A PLACEMENT METHODOLOGY FOR LOW POWER VLSI CIRCUITS

By

Manuel A. Jiménez-Cedeño

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Electrical and Computer Engineering

1999

ABSTRACT

A PLACEMENT METHODOLOGY FOR LOW POWER VLSI CIRCUITS

By

Manuel A. Jiménez-Cedeño

Digital VLSI design techniques have been traditionally centered around optimizing circuit speeds and/or area requirements. This has created an ever-shrinking, ever-faster trend in circuit technology enabling the proliferation of powerful portable electronic devices. Unfortunately, there has not been a similar growth in the battery technology to accompany these circuits. This fact, among others, has shifted traditional design objectives to also include as a major goal the minimization of power consumption. Numerous approaches have been reported at the architectural and logic levels. The physical design level however, has not received that much attention leaving a gap in the design cycle of digital circuits.

The objective of this dissertation is to partially fill this gap through the development of a new placement methodology aimed at reducing the power dissipation of macro block-based layouts. An optimization heuristic based on simulated annealing was developed with the goal of reducing the switched capacitance of signal nets in the circuit while maintaining high quality in terms of area and speed. Results from the application of this methodology show that power reductions exceeding 15% can be obtained without compromising other aspects of the layout quality. Several contributions have emerged from this research in addition to an effective power reducing placement tool. These include efficient new algorithms in the areas of graph theory, computational geometry, and physical design.

Copyright © by

Manuel A. Jiménez-Cedeño

1999

To my wife Nayda and our son Victor Manuel

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Michael Shanblatt for his savy guidance during my years as a Ph.D. student, especially during the most critical times of this research. I would also like to thank the members of my graduate committee, Dr. Diane Rover, Dr. Richard Enbody, Dr. Antony Wojcik, and Dr. Dennis Gilliland for their help in the revision and correction of this manuscript.

I would also like to thank my wife Nayda Santiago for her unconditional love and support and to our son Victor Manuel for all the happiness he has brought into our lives. I am also grateful to my parents for their strong support to help me achieve all the objectives I have set on my life.

Thanks to mi friends and colleages Ziad, Andres, and Ron for their help whenever I needed it.

TABLE OF CONTENTS

LIST OF TABLES viii

LIST OF FIGURES ix

1 Introduction 1
- 1.1 Motivation and Objective . . . 2
- 1.2 General Approach . . . 3
- 1.3 Dissertation Overview . . . 5

2 Previous Work 7
- 2.1 Power Dissipation Mechanisms in Digital CMOS Circuits . . . 8
- 2.2 Power and Switching Activity Estimation Techniques . . . 11
  - 2.2.1 Statistical Methods . . . 12
  - 2.2.2 Probabilistic Methods . . . 17
  - 2.2.3 Comparative Analysis and Selection of Estimator . . . 35
- 2.3 Power Minimization Techniques . . . 38
- 2.4 Placement Techniques . . . 43
  - 2.4.1 Wirelength Estimation . . . 47
  - 2.4.2 Partition-based Techniques . . . 50
  - 2.4.3 Force-directed Methods . . . 52
  - 2.4.4 Genetic Placement Techniques . . . 53
  - 2.4.5 Simulated Annealing . . . 55
  - 2.4.6 Other Approaches . . . 60
- 2.5 Summary . . . 61

3 Problem Formulation 63
- 3.1 Terminology and Definitions . . . 63
- 3.2 Problem Statement . . . 64
- 3.3 Approach . . . 67
- 3.4 Summary . . . 71

4 New Steiner Tree Heuristics for Physical Design Problems 72
- 4.1 Why more MRST heuristics? . . . 72

4.2 Terminology and Definitions . . . 75
4.3 Layout Graph Extraction and Reduction . . . 77
4.4 Median Biased Trees . . . 79
4.4.1 Median Biased Heuristic (MBH) . . . 80
4.4.2 Diameter Biased Heuristic (DBH) . . . 82
4.5 Experimental Results . . . 84
4.6 Summary . . . 89

5 Implementation of the Power Optimizing Placement Tool 90
5.1 Input Format . . . 90
5.2 Internal Layout Representation . . . 91
5.3 Annealing Algorithm . . . 93
5.3.1 Cost Function . . . 93
5.3.2 Perturbation Function . . . 110
5.3.3 Annealing Schedule . . . 115
5.4 Summary . . . 122

6 Results and Analysis 124
6.1 Circuit Profiles . . . 124
6.2 Comparative Analysis . . . 125
6.2.1 Wirelength Results . . . 126
6.2.2 Switched Length Results . . . 128
6.2.3 Other Results . . . 130
6.3 Asymptotic Behavior . . . 131
6.4 Summary . . . 133

7 Conclusion 135
7.1 Research Summary . . . 135
7.2 Contributions . . . 137
7.3 Future Work . . . 140
7.4 Closing Remarks . . . 141

A Using the SAPPO program 142
A.1 Command Line Syntax . . . 142
A.2 Input Format . . . 143
A.2.1 The LPY Netlist Format . . . 144
A.2.2 Rules and Constraints Specifications . . . 153
A.3 Output Files . . . 153
A.4 Support for Other Netlist Formats . . . 154

BIBLIOGRAPHY 155

LIST OF TABLES

2.1 Summary of reviewed estimation techniques. 37

4.1 Results of graph reduction algorithms. 86
4.2 Running time of the heuristics in ms. 88

5.1 Effects of the displacement to exchange ratio. 111
5.2 Temperature reduction rule. 117
5.3 Effects of the temperature reduction rate. 118
5.4 Parameters controlling the convergence rate. 119

6.1 Characteristics of the circuits used in the evaluation of results. 125
6.2 Wirelength optimization for the SAPPO tool and a reference heuristic. . . . 127
6.3 Additional results reported on the set of benchmark circuits. 127
6.4 Running times for the SAPPO program. 128
6.5 Wirelength results versus switched length on the set of benchmarks. 129
6.6 Improvements obtained when switched length was used as target. 129
6.7 Characteristics of scaled layouts used for asymptotic behavior tests. 133
6.8 Results of suboptimality tests. 133

LIST OF FIGURES

2.1 Sources of power dissipation in a CMOS inverter. 9
2.2 Input voltage and short-circuit waveforms. 9
2.3 Glitch activity caused by gate delays. 23
2.4 Conventional FSM. 25
2.5 FSM transformation to capture temporal correlations. 25
2.6 Design flow of digital integrated circuits. 44
2.7 Basic layout styles. 46
2.8 Wirelength measures produced using different estimation methods. 51

3.1 Circuit to be placed with estimated switching activities per net. 67
3.2 Placement solutions with objective functions for: a) wirelength, b) switched length. 68
3.3 Relationships among tasks in the solution approach. 69

4.1 Switching activity *vs.* number of terminals in selected circuits. 74
4.2 Macro block layout and its equivalent graph. 77
4.3 Reduced layout graph of Figure 4.2. 79
4.4 Proportion of best solutions given by each heuristic. 87
4.5 Proportion of worst solution given by each heuristic. 87

5.1 Sample layout with augmented dimensions. 96
5.2 Cost behavior with early penalty removal. 98
5.3 Cost behavior with late penalty removal. 98
5.4 Scan of non-rectagular modules. 105
5.5 Space exploration performed by the perturbation function. 114
5.6 Solution cost vs. temperature for fixed and variable values of v. 118
5.7 Effects of the convergence rates. 121
5.8 Stability of the annealing schedule. 122

6.1 Placement of benchmark `hp`. 130
6.2 Base and scaled circuits used for asymptotic tests. 132

CHAPTER 1

Introduction

Very large scale integration (VLSI) design techniques for digital circuits have been traditionally centered around optimizing circuit performance, area requirements, or both. This has created a trend in circuit technology where about every eighteen months new, faster, and smaller devices are developed, enabling the proliferation of powerful portable electronic products. Unfortunately, there has not been a similar growth in the technology of the batteries used for feeding these circuits. This fact, combined with circuit reliability concerns, the packaging and cooling costs associated with power hungry devices, and environmental issues, have been gradually shifting the traditional objectives in digital VLSI design to also include as a major goal the minimization of power consumption.

A growing number of schemes have been developed in the last six or seven years targeting different issues related to the problem of optimizing the power dissipation in digital circuits. Working in the same direction, this dissertation presents the development and evaluation of a new placement methodology aimed at producing digital VLSI circuit layouts with reduced power dissipation.

This chapter has been organized as follows. Section 1.1 presents a discussion of the motivation leading to this research and establishes the central objectives. A description of the approach followed is outlined in Section 1.2. The final section provides an overview of the remainder of this dissertation.

1.1 Motivation and Objective

As was introduced above, during the last few years the reduction of power dissipation in VLSI integrated circuits has increasingly become an issue of concern. As the attention of the research community has gradually shifted to search for power efficient design mechanisms, a growing number of approaches have been reported which are directly or indirectly related to the low-power objective.

A considerable amount of effort has been invested in the development of accurate and time-efficient power estimation tools [1–6]. Numerous techniques have also been reported aiming at the design of low power devices by working at different levels of the design hierarchy [7–12].

While most of these approaches have focused their efforts on the savings attainable at either the architectural or the logic synthesis levels, only a few of them have considered the implications of the physical design process [13, 14]. This research pattern not only has neglected the potential savings in power which could be obtained by methods targeting the physical level, but has also failed to provide an undisrupted design sequence to be followed when an application needs to be designed from beginning to end with a low power objective in mind.

An objective of this dissertation is to narrow the gap existing in low power design techniques between the stages of logic design and IC fabrication with the development of a mechanism for reducing the power dissipation of circuits during their physical design process. More specifically, the development of a placement methodology for macro block-based layouts is presented, whose objectives include reducing the power dissipation of the resulting layout while maintaining the quality in terms of area and delay similar to that obtained with non-power conscious techniques.

The selection of the placement problem as the targeted physical design stage and macro block-based layouts as the semi-custom integration technology for its application, was made for the following reasons. Placement is the most general and crucial stage of the entire physical design process. It crucially affects the routing layout stages which it precedes. A

successful routing can only be completed if it starts from a quality placement. Moreover, with the development of a placement methodology, the foundation of a number of other physical design tools are created.

The macro block-based layout style is a semi-custom integration technology closest to the ideal full-custom style and contains a high degree of freedom in the placement process. This means that by introducing into a macro block-based placement methodology the appropriate set of constraints, the placement of standard cells, gate arrays, and mixed technologies can be treated as special cases.

1.2 General Approach

To successfully achieve the objectives set for this dissertation, a multi-stage research approach was followed. This approach covered several steps, starting from an exhaustive search of background information to an extensive evaluation process verifying the correct and unified performance of the diverse mechanisms developed to support the proposed methodology.

To properly target a power reduction strategy for digital circuits it was essential to understand the mechanisms of their power dissipation behavior, the ways in which they can be predicted, and the effective mechanisms which had been developed to successfully exert control on them. Also, the creation of a set of efficient placement rules could only be reached through an extensive knowledge of all successfully attempted approaches. Thus, the first stage of this dissertation consisted of a detailed study these areas. Outcomes derived from this stage include a classification of power estimation mechanisms and an assurance of the relevance and validity of the subject of this dissertation.

The next stage included an analysis of methods used for finding minimum interconnection networks and their effect in low-power placement and routing applications. Reasons were found justifying the development of improved heuristics for these tasks. The outcome of this stage included two new efficient heuristics for the minimum rectilinear Steiner tree problem with potential applications in wirelength estimation and global routing.

In the next stage, the set of benchmarks circuits to be used for evaluating the developed tools were selected. A set of circuits from the MCNC*benchmark suite were chosen for this purpose. The formats in which these circuits were available did not support the direct integration of parameters for low-power physical design. This made it necessary to develop extensions to an existing format to support the required power dissipation parameters. The outcome of this stage was a new netlist format supporting power optimizing physical design tools.

At this point, the most relevant aspects of the methodology regarding power control parameters, wirelength estimators, optimization engine, and the target layout style were clearly defined. Therefore, a detailed implementation of a placement tool supporting the methodology was the next step. For its completion, a simulated annealing-based optimization core, a novel overlap detection mechanism, and several wirelength estimation options were implemented. These components were merged together to assemble a placement tool whose objective function was aimed at reducing the switched length of the networks in the circuit being placed. The switched length was used as a direct measure of the power dissipated by the signal networks in the layout. The tool operates on the enhanced physical netlist format supporting power parameters developed earlier. An additional outcome from this stage was the development of improved algorithms for the manipulation of general rectilinear shapes.

A stage of careful calibration of the annealing schedule, the perturbation function that allows exploration of the solution space, and the balance of weights in the cost function was required to produce stable and consistent behavior of the annealing process. A series of tests were performed to assess the consistency and stability of the tool. The outcome of this stage was the creation of a stable and consistent placement tool embedding the power minimization methodology of this dissertation.

A final stage of analysis and evaluation followed. This stage included the validation of the placements produced with the implemented tool and an assessment of power reduction

*Microelectronics Center of North Carolina

achieved with the developed methodology. Outcomes of this process indicate that a tool producing quality layouts was implemented and that the desired power reductions were achieved by the methodology without compromising the quality of the layouts in terms of area or delays.

1.3 Dissertation Overview

The rest of this document has been organized as follows. Chapter 2 discusses the mechanisms involved in the power dissipation of digital circuits. This discussion identifies the set of controllable parameters that define the degrees of freedom available in the low power design space. Reviews of background work reported in the areas of power estimation, power reduction, and layout placement are also presented in this chapter, highlighting the trends followed in each of them.

The problem of placing macro block-based layouts for low power dissipation is formally stated in Chapter 3. This chapter also includes an overview of the approach followed in finding a solution to the stated problem. This overview serves as a map to the implementation of the methodology under discussion.

Chapter 4 introduces the median-biased heuristic (MBH) as an alternative mechanism to find approximate solutions to the rectilinear Steiner minimum tree problem in graphs. Also in this discussion, several graph reduction mechanisms relevant to the RMST problem are proposed. Test results highlighting the superior characteristics of the proposed techniques are presented. The potential of these techniques for improving mechanisms in low-power physical design is also discussed in this chapter.

The implementation of the placement methodology central to this dissertation is detailed in Chapter 5. It discusses the design of the simulated annealing algorithm, including the details on how the calibration of the annealing schedule was performed. In the frame of the discussion of functions for evaluating the cost of layout placements, new algorithms in the area of computational geometry are introduced and discussed. A series of tests presented at the end of this chapter demonstrates the stability and consistency of the algorithms

supporting the methodology.

The results of applying the developed placement tool and power reduction methodology on a set of standard benchmark circuits are the subject of Chapter 6. A comparative analysis of wirelength optimizations is used to validate the placement tool, showing why its results can be considered among the best reported for the benchmark test cases. An assessment of the power reduction results is presented next, showing the effectiveness of the placement methodology as a mechanism to reduce the power dissipation of physical layouts. The chapter concludes with an analysis of the asymptotic behavior of the methodology and tool, projecting the quality of its results to large circuits.

The last chapter summarizes the central ideas of this dissertation, highlighting the contributions that emerged from this effort, and pointing to several research directions which could further enhance the achievements made with this work.

CHAPTER 2

Previous Work

The problem of minimizing the power dissipation of digital VLSI circuits has become a target of increasing attention in the last years. Numerous techniques have been reported which propose diverse solutions to this problem. This chapter presents a review of salient previous work reported in this area, with emphasis on those more closely related to the areas of contribution of this dissertation.

To properly target a power reduction strategy in digital CMOS circuits it is essential to understand the mechanisms of their power dissipation. The first section of this chapter provides a background discussion of this subject. The second section reviews power and switching activity estimation techniques which are central to any power reduction mechanism. This discussion also provides the basis for the selection of the estimator used in this research. Several representative power reduction mechanisms are discussed next, revealing the pattern followed in reported strategies. Section 2.4 presents layout placement techniques, reviewing methodologies used in the specific stage of the design process where the chosen power reduction mechanism fits. The last section summarizes the findings made upon completion of the literature review process, explaining some important decisions made regarding the discussed subjects and highlighting the relevance of the subject of this dissertation.

2.1 Power Dissipation Mechanisms in Digital CMOS Circuits

The sources of power dissipation in digital circuits are frequently classified into *static* and *dynamic* components, according to the type of current circulation from which they are produced [15].

The power component due to the static currents, as illustrated in Figure 2.1a, essentially arises from two types of sources: i) the reverse bias leakage in the junction of the parasitic diodes formed between the diffusion regions and the substrate in the transistors; and ii) the subthreshold conduction or substrate injection current in the device. In well designed circuits, leakage currents are on the order of nanoamperes and subthreshold conduction is typically of the order of a few tens of microamperes. Adding these components in a single term I_{leak}, their contribution to the total power dissipation of the circuit can be represented by $P_{sc} = I_{leak} \cdot V_{dd}$. This term is typically less than a 1% of the total dissipation, a fraction which can be neglected in most practical analyses.

The dynamic component of the power dissipation is caused by the transient behavior of digital devices upon signal transitions. Typically, two types of dynamic currents are identified: the *short-circuit* and the *capacitance charge-discharge* currents.

The short-circuit current is a current spike originating during input signal transitions due to a short period in which the transistors in the path from V_{dd} to ground are all in conduction. This effect is illustrated in Figure 2.1b for a typical CMOS inverter.

Assuming an unloaded circuit ($C_L = 0$) under an input waveform as in Figure 2.2, the short circuit current circulates when the value of the input voltage V_{in} is below $V_{dd} - |V_{tp}|$ and above V_{tn}. The levels V_{tp} and V_{tn} in Figure 2.2 represent the threshold voltages for the p and n transistors respectively and β their current gains. The terms t_r and t_f represent the rising and falling times respectively, and t_p the period of the input waveform.

The average short circuit current in a single node is frequently represented as $I_{sc} = f_{clk} \cdot Q_{sc}$, where f_{clk} represents the switching frequency and Q_{sc} the amount of charge transferred in each transition. The term Q_{sc} is a strong function of the threshold levels, the voltage swing, and other physical characteristics of the device (see [16] for a comprehensive

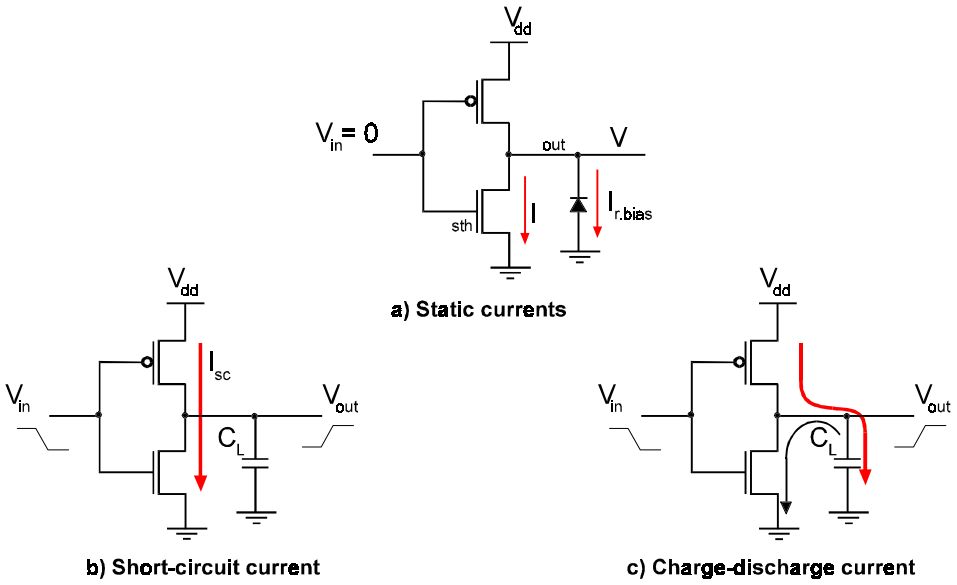


Figure 2.1. Sources of power dissipation in a CMOS inverter.

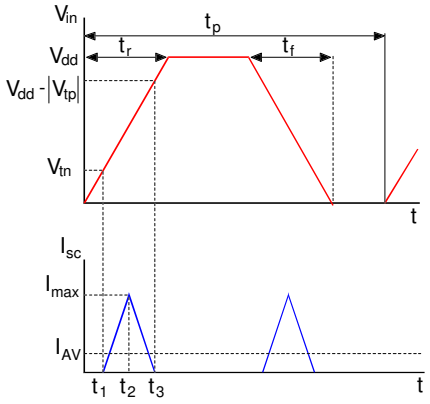


Figure 2.2. Input voltage and short-circuit waveforms.

discussion of this subject). The expression for the portion of the power dissipation caused by the short circuit current P_{ds} is shown in equation (2.1). This portion is typically around 10% of the total power dissipation in well designed CMOS circuits.

$$P_{ds} = \alpha \cdot f_{clk} \cdot Q_{sc} \cdot V_{dd} \tag{2.1}$$

The term α in equation (2.1) represents the expected number of transitions per clock cycle performed by each node in a CMOS circuit. This value, commonly called the *switching activity factor* [17], is highly dependent on the input streams applied to the circuit and the logic equation relating the node function to the primary inputs.

The second type of dynamic current is originated by the charge-discharge of the equivalent capacitance C_L at the output of a net driver. This effect is illustrated in Figure 2.1c with a basic CMOS inverter. It can be seen that when the output of the inverter switches high, the load capacitance is charged by a current $i(t) = C_L \frac{dv(t)}{dt}$. In the next change of the output, *i.e.* when it switches low, the charge stored in C_L is transferred to ground. The net result of this process is a circulation of the current $i(t)$ from V_{dd} to ground. If the node switches at a frequency $\alpha \cdot f_{clk}$ in Hz., then its power dissipation due to this effect in each transition will be given by equation(2.2).

$$P_{dl} = \frac{1}{2} \cdot \alpha \cdot f \cdot C_L \cdot V_{dd}^2 \tag{2.2}$$

This last component, when taken over all nodes, is typically responsible for about 90% of the total power dissipation of digital CMOS circuits. Combining the equations for P_{sc}, P_{ds}, and P_{dl} over an entire circuit with r nodes, and considering that load capacitances and switching activity values are, in general, different among nodes, the expression for the total power dissipation of the circuit can be written as:

$$P_T = \frac{1}{2} V_{dd}^2 \cdot f_{clk} \sum_{k=1}^{r} (\alpha_k \cdot C_{Lk}) + V_{dd} \cdot Q_{sc} \cdot f_{clk} \sum_{k=1}^{r} \alpha_k + I_{leak} \cdot V_{dd} \tag{2.3}$$

Neglecting the third term of the addition in equation (2.3), it can be seen that the power

dissipation in CMOS circuits is essentially a function of the operating frequency f_{clk}, the supply voltage V_{dd}, and the individual load capacitance C_{Lk} and switching activity α_k of each node. Given that the speed performance of the system is a direct function of f_{clk}, from the above parameters, only the last three, namely V_{dd}, C_{Lk}, α_k and are typically addressed by most power reduction strategies.

2.2 Power and Switching Activity Estimation Techniques

Power estimation deals with the problem of assessing the average power dissipation of a circuit at some point during its design process. As can be appreciated from equation (2.3), power dissipation in digital circuits is a strong function of the switching activity of internal nodes. But switching activity is an input-dependent parameter. This causes a given circuit to exhibit different power dissipation behaviors for different sets of input patterns. This dependence is one of the factors that makes the problem of power estimation a particularly difficult task.

Early estimation techniques were strongly dependent on input patterns [18]. They were based on pure logic or circuit level simulation and usually required complete and specific information about the input signals. Despite the capability of these methods of yielding very accurate estimates regardless of technology, design style, functionality, *etc.* nowadays they are seldom used because they become unfeasible as the gate count, the number of signals, and the length of input vectors increases.

To cope with these difficulties, recent techniques, in most cases, rely on statistical or probabilistic methods to somehow capture the behavior of the circuit upon typical input patterns and speed-up the power estimation process. According to the underlying mechanism employed, these approaches can be classified into two broad categories: *statistical techniques*, sometimes also called simulation-based approaches and *probabilistic techniques*.

Statistical approaches rely on logic or circuit simulation to provide an estimate. Simulations are stopped when the internal activities are considered to have converged to their average value. Criteria like the Monte Carlo method are employed to limit the conver-

gence of switching activities to some user-specified accuracy and confidence levels with the purpose of reducing the simulation time.

Probabilistic estimation uses stochastic techniques to characterize the behavior of circuit nodes upon input vectors. These techniques usually require user-specified parameters that characterize the typical behavior of the primary inputs in a probabilistic form. These parameters are then propagated to all internal and output nodes of the circuit.

A discussion of selected statistical and probabilistic estimation techniques follows. But, before entering into their details a few commonly used terms are defined.

Definition 2.1 (Signal Probability $P_s(x)$) *The probability that the signal at node x evaluates to logic one [19].*

Definition 2.2 (Transition Probability $P_t(x)$) *The probability that the logic signal at node x experiences a change in its logic state.*

Definition 2.3 (Spatial Correlation) *Two logic signals x and y are spatially correlated if in the same time slot the state of one of them depends on the state of the other.*

Definition 2.4 (Temporal Correlation) *The dependence of the current state of a signal on its previous logic value.*

Definition 2.5 (Glitch Activity) *The portion of the switching activity caused by glitches when the model used to represent the circuit has a delay other than zero.*

Definition 2.6 (Transition Density) *The average number of transitions per unit time experienced by a circuit node [20].*

2.2.1 Statistical Methods

Statistical methods are simulation-based approaches where statistical techniques are applied to reduce the number and length of the simulation runs required to perform an accurate estimation. These methods require the creation of a sample space with characteristics which allow the application of statistical tools. The characteristics of the sample space,

which usually include ergodicity and stationarity of the underlying random process, are manifested differently for sequential and combinational circuits. For this reason, most of the reviewed techniques explicitly identify the kind of circuits to which they are applicable.

Burch, *et al.* proposed a method using Monte Carlo simulation to estimate the total power consumption in combinational circuits [4]. Their method consisted of applying randomly generated input patterns to the circuit while monitoring the resulting power dissipation with a simulator. The average power dissipation was obtained by applying statistical mean estimation on the collected samples. This approach assumes that the distribution of power dissipation over a period T is very close to a normal. The normality property is used to develop a stopping criterion for the simulations when computations are close enough to the average power. The criterion is built as follows: Let η_T and S_T be the sample average and sample standard deviation of the random variable P_T (total power) obtained from N simulations of time length T each. Then with confidence $(1-\alpha)\times 100\%$ it can be said that $|\eta_T - E(P_T)| < t_{\frac{\alpha}{2}} \frac{S_T}{\sqrt{N}}$, where $t_{\frac{\alpha}{2}}$ is obtained from the $t-$distribution with $N-1$ degrees of freedom [21]. Thus, for a desired percentage of error ε in the power estimation with a confidence level $(1-\alpha)$ the circuit must be simulated

$$N \geq \left(\frac{t_{\frac{\alpha}{2}} S_T}{\eta_T \varepsilon} \right)^2 \tag{2.4}$$

number of times, where T, the length of each simulation, is determined experimentally.

This approach was found to be very effective for combinational circuits where the normality assumption held. For circuits with nonnormal distribution in their power dissipation the stopping criterion erroneously terminated the simulation, leading in most cases to significant errors in the estimates. The algorithm does not work for sequential circuits since it has to wait for a setup time, $T_{\max}$, until the internal transition process in the circuit becomes stationary. In a combinational circuit this time is typically equal to the length of the longest delay path in the circuit, a criterion difficult to apply to circuits with feedback. The results obtained from the application this technique on circuits from the ISCAS 85 benchmark, specifying 5% error and 99% confidence level, were compared to those of a probabilistic

technique [20] and a logic simulation. By using the logic simulation as a reference, it was found that in most cases, the error produced by the Monte Carlo technique was less than that of the probabilistic technique, at the expenses of a longer execution time.

Xakellis and Najm extended the concept presented in [4] to also provide an estimate of the power dissipation in internal nodes with user-specified accuracy and confidence [22]. This extended technique classified the internal circuit nodes into *regular* and *low-density*, according to their sampled transition densities. The purpose of such a classification was to modify the stopping criterion for low-density nodes by the application of an absolute error bound. This bound was defined by $\varepsilon D_{\min}$, where ε and $D_{\min}$ were user-specified parameters indicating the accuracy and threshold value of the transition density used for performing the classification. In this way, the approach limited the otherwise large value of N required by nodes with infrequent activity to reach convergence. The relaxation in the error bound for low-density nodes was reported to have little effect in the accuracy of the global estimation, mainly because of their small contribution to the total power dissipation. This technique however, inherited most of the characteristics of its predecessor by assuming normality in the distribution of the node transition densities and an exclusive application to combinational circuits. Two additional characteristics were found in this approach. First was the nonuniformity of the estimation error in the internal nodes; and second, that execution times and accuracy of the estimates were highly sensitive to the value of $D_{\min}$ specified by the user. Execution times were reported in hours for several reported test cases.

Najm, *et al.* proposed a Monte Carlo technique for estimating the switching activity in the latch outputs of a finite state machine (FSM) [2]. By applying randomly generated input sequences to a high-level functional description of the circuit under a zero-delay model, statistics on the latch outputs were collected. These statistics were then fed to a combinational power estimation tool which estimated the power dissipation of the whole circuit. The Monte Carlo technique was applied in a similar way as in [22] for determining N, the number of runs required to produce a sample space of the node activities. In such a

space, the signal probabilities and transition densities of the latch outputs were estimated with a user-specified accuracy and confidence level.

To overcome the problem of determining the setup time required before starting the collection of samples, two sets of simulations of the same FSM were performed. Each of these sets of simulations started from different initial states while recording the average and difference of their latch probabilities. When the recorded values had remained in a window $\pm\varepsilon$ for three consecutive clock cycles, the probabilities were said to have converged. The choice of three time instants was arbitrary and user modifiable. This estimation approach reported results for very large FSMs (up to 1452 flip-flops). However, the execution time required was considerably large. In order to produce estimates within a reasonable time, 31 parallel simulations of a given FSM had to be performed each time until accumulating about 500 runs, which was the average value of N necessary for creating an appropriate sample space where estimates can be made with 5% error and 95% confidence level. The reported results did not indicate if the reported execution times corresponded to the estimation of the latch statistics only or if they also included the time required for the estimation in the combinatorial part.

Another Monte Carlo approach for estimating switching activity in sequential circuits was proposed by Stamoulis, based on the analysis of paths in the state transition graph (STG) of FSMs [23]. A path in a STG was defined as a sequence of adjoining transitions, starting and ending at two states identified as *path head* and *path tail* respectively. This statistical method assumed that state probabilities after power up were uniformly distributed, that latch outputs were glitch free, and all latches reached steady state before their next state was allowed to enter into the combinational logic. Based on the concept of initializable circuits, it was established that the random process describing the state of the latch outputs was stationary and states which were n clock cycles apart became independent as $n \to \infty$, also implying ergodicity. The stationary and ergodic properties of the process led to the conclusion that the transition probabilities at the latch outputs converge to an average which was independent from the initial state. In order to obtain an estimate of

such averages the circuit was simulated on different paths by selecting equiprobable initial states.

The estimation process was performed in two steps. First, the pathwise transition probabilities were estimated using the Monte Carlo method to determine the minimum number of path samples required to have an error ε with confidence $(1-\alpha)\times 100\%$. Normality was assumed. The second step averaged the probabilities obtained over all the paths to make the estimate of the switching probabilities. Nodes with low average probabilities were allowed to have a larger estimation error to keep the value of N from growing too large. In each path, the probabilities were said to have converged if the difference between the minimum and maximum values over all the runs was less than a preset threshold value. The results obtained using the sequential circuits in the ISCAS 89 benchmark indicate that 89% of the FSMs satisfied the initializability condition. Results for 5% error and 95% confidence were presented, showing reasonable running times for both sequential and combinational parts of the circuits.

A nonparametric statistical technique for estimating the total power dissipation in combinational circuits was proposed by Yuan, *et al.* [24]. Their technique made no assumption about the distribution of the power dissipation of a digital circuit. It assumed that the circuit inputs were mutually independent, and that the probability density function (pdf) of the power distribution could be approximated by a continuous function. The application of the Kolmogorov-Smirnov theorem [25] enabled an estimation of the limiting distribution of $\sqrt{n}D_n$, where D_n was the maximum difference between the sampled and real cumulative distribution functions (cdf) of the power dissipation. The value of D_n was taken when n, the number of samples, was sufficiently large, using an upfront specified confidence level. This allowed the establishment of a confidence band for the cdf of the power dissipation. In order to develop a stopping criterion for the simulations, the confidence band was used to find the bounds on the power dissipation as follows.

Let P be a random variable of the circuit power dissipation. Then μ_p , the average value of P, is $\int_{-\infty}^{\infty} pf(p)dp$, where $f(p) = dF(p)/dp$ and $F(p)$ is the cdf of P. Since $F(p)$

is a nondecreasing function, it has a well defined inverse $F^{-1}(p)$. Therefore, introducing a change of variables $u = F(p)$, it can be written $\mu_p = \int_0^1 F^{-1}(u)\,du$. If the observed value of D_n is represented by c, then the stopping criteria for a maximum error ε can be written as

$$\max\left[\frac{\overline{\mu_p} - \mu_{pL}}{\overline{\mu_p}}, \frac{\mu_{pU} - \overline{\mu_p}}{\overline{\mu_p}}\right] \leq \varepsilon, \tag{2.5}$$

where $\mu_{pL} = \int_0^1 F_n^{-1}(u-c)\,du$, $\mu_{pU} = \int_0^1 F_n^{-1}(u+c)\,du$, and $\overline{\mu_p}$ is the sample mean obtained from n independent simulations. In the above equations $F_n(p)$ is an approximation of the cdf of the power distribution constructed from the n random samples of P obtained from the simulations. This approximation was found to introduce errors no larger than 1.2% when compared to the results of a simulation based on one million random samples. The results obtained by applying this method to a set of benchmark circuits, mostly from ISCAS 85, demonstrated that the technique can produce accurate estimates of the total power dissipation, regardless of the specific distribution exhibited by the circuits. These included several circuits analyzed by Burch *et al.* and found to have distributions other than normal [4]. The results, which included estimates for very large circuits, did not indicate the time requirements for completing the estimates.

2.2.2 Probabilistic Methods

The use of probabilities to characterize the random behavior of input vectors in digital circuits was initially proposed for the generation of test patterns for fault detection [26]. The concept was later extended to handle the problem of estimating the power dissipation of digital CMOS circuits. The probabilistic estimation of power dissipation requires the consideration of three main factors which deeply affect the behavior of signal transitions in digital circuits.

The first factor is the delay model used in the circuit representation. Under a zero-delay model, internal and output nodes perform a single transition per each single change in an input signal. This implies that under proper assumptions, estimates performed using a zero-delay model will yield a lower bound on the switching activity of a circuit. When either

a unit- or real-delay model is used, internal and output nodes might experience several transitions triggered by a single input transition. Either of these two models will allow for the effect of glitches in power estimates, the real-delay model being the most accurate.

The second factor of importance refers to the form in which signal correlations will be considered in the primary inputs and internal nodes. Most probabilistic techniques assume that external input signals are independent, which is a valid model for a wide range of circuits. However, even when this condition is true, spatial correlations are created on internal nodes by the reconvergent fanout problem. Some estimation techniques have been proposed neglecting these correlations. Unfortunately, reconvergent signals are common in digital circuits and assuming independence in internal signals usually leads to significant estimation errors.

A third factor to consider is the type of circuit analyzed, *i.e.* whether the evaluation is performed on a combinational or sequential circuit. Although a wide range of combinational circuits satisfy the independence assumption on its external inputs, such an assumption is not valid for the combinational block implementing the next state logic in a sequential circuit. The inputs to such a block in a typical Mealy machine include the present state (PS) and primary inputs (PI). In most cases the PIs hold with independence premise. However, the PS inputs exhibit high temporal correlations induced by the state transition behavior of the machine and at the same time considerable spatial correlations created by the state encoding scheme. The way in which these three factors are considered creates essential differences in the techniques discussed below.

The application of probabilistic measures to characterize the random behavior of signals in digital circuits was firstly proposed by Parker and McCluskey [19]. They developed basic definitions and expressions related to the assignment of probabilities to logic signals and proposed two algorithms for propagating these probabilities from PIs to internal and output nodes in combinational circuits. These algorithms assumed independence in the primary inputs and required the knowledge of their signal probabilities.

One of the pioneering works to propose the use of probabilities to estimate the power

dissipation of digital circuits is due to Cirit [27]. This methodology assumed a zero-delay model and spatiotemporal independence of signals when propagating the input probabilities through the circuit. The propagation algorithm worked at the transistor level but could be easily tailored for providing estimates at the gate level. The spatial independence assumption made in the propagation of the probabilities introduced considerable errors in the estimates it produced.

A widely known measure of switching activity in digital circuits is due to Najm, who proposed the *transition density* as a measure of the average switching rate at a circuit node [6, 20]. Representing by $n_x(T)$ the number of transitions of the signal $x(t)$ at node x in the time interval $\left(-\frac{T}{2}, +\frac{T}{2}\right]$, the transition density at x, denoted $D(x)$, was defined as

$$D(x) = \lim_{T \to \infty} \frac{n_x(T)}{T}. \tag{2.6}$$

This measure allowed for estimating the switching activity in a large period for general signals without the requirement of a periodic pattern. Note that equation (2.6) can be incorporated into equation (2.3) replacing the term $\alpha_k \cdot f_{clk}$. In order to estimate the power dissipation of a circuit, the transition densities and equilibrium probabilities at the external inputs have to be specified by the user. For an input signal x, the value of $D(x)$ can be obtained using equation (2.6), while the equilibrium probability is computed as $P(x) = \frac{1}{T}\int_{-T/2}^{+T/2} x(t)\, dt$. These values are then propagated to the internal and output nodes as follows. Given a circuit with n spatially independent inputs $x_i(t)$, $i = 1 \ldots n$, for which the values $D(x_i)$ and $P(x_i)$ are known, the transition density at any internal or output node $y_j(t)$ can be obtained as

$$D(y_j) = \sum_{i=1}^{n} P\left(\frac{\partial y_j}{\partial x_i}\right) D(x_i). \tag{2.7}$$

In the above equation, $P(\cdot)$ represents the signal probability of the expression in parentheses and $\frac{\partial y}{\partial x}$ represents the Boolean difference of y with respect to x. The Boolean difference of

a function $y = f(X), X = x_1, \ldots, x_n$ with respect to signal x_i is defined as

$$\frac{\partial f(X)}{\partial x_i} = f(X)|_{x_i=1} \oplus f(X)|_{x_i=0}, \tag{2.8}$$

where $\oplus$ represents the exclusive-or (XOR) operation.

Although the definition of the transition density was capable of accommodating the activity created by glitches in a nonzero-delay model, the propagation algorithm proposed by Najm was insensitive to the module delays and therefore worked on a zero-delay representation of the circuit. The applicability of the algorithm as proposed was limited to combinational circuits. One of its major drawbacks was the spatial independence requirement in the module inputs. Although this requirement might be met by the primary inputs to the circuit, reconvergent signals in the inputs of internal modules destroy this property, introducing errors in the computations. By analyzing the results reported in [20], it can be found that 60% of the circuits analyzed had average estimation errors larger than 20%. The magnitude of these errors could be reduced by making the internal blocks large enough to enclose as many reconvergent nodes as possible. This approach however, has a high cost in the space and time complexity of the propagation algorithm since the size of the binary decision diagrams (BDDs) [28] it works on are exponential in the number of inputs feeding the blocks.

The original model proposed for propagating the transition density was later extended by Najm to include the filtering effect of the inertial delay in logic gates [29]. The justification of such an extension was that in the original model, the transition densities might be severely overestimated in high frequency applications because it did not limit the minimum pulse width which could be propagated through a logic gate. In order to model this effect, a delay block with low-pass filtering capabilities was attached to every module output. In this block, a transition was transmitted from input to output with a delay τ_i, $i = 0, 1$ if the input did not change during that time. The length of the delay inserted would be τ_0 or τ_1 depending on whether a $0 \rightarrow 1$ or $1 \rightarrow 0$ transition was being propagated. In order to include this block in the probabilistic propagation of the transition density, a stochastic

model for the filtering block itself was developed.

To develop this model it was assumed that the cumulative distribution functions $F_0(t)$ and $F_1(t)$ of the low and high pulses transmitted by the block were exponential. After analyzing the filter behavior and introducing some simplifying assumptions about the pulses at the filter input, the expressions obtained for propagating the equilibrium probabilities and transition densities through the filter block were

$$\begin{aligned} P(y) \simeq\ & P_F \times \left[\frac{1}{1 - F_0(\tau_0)} - P(x) \right. \\ & \left. + \frac{(\mu_1 - \tau_1) + \tau_1 F_1(\tau_1)/2}{1 - F_1(\tau_1)} - \frac{(\mu_0 - \tau_0) + \tau_0 F_0(\tau_0)/2}{1 - F_0(\tau_0)} \right] \end{aligned} \tag{2.9}$$

and

$$D(y) = P_F \times D(x), \tag{2.10}$$

where x and y were assumed to be the filter input and output respectively, μ_0 and μ_1 the average lengths of the low and high pulses at the filter inputs, and P_F a relation between the distributions of the high and low pulses given by

$$P_F = \frac{[1 - F_0(\tau_0)][1 - F_1(\tau_1)]}{1 - F_0(\tau_0) F_1(\tau_1)}. \tag{2.11}$$

The results obtained indicate that the inclusion of the low pass filter in the module outputs attained the goal of reducing the overestimation error of the transition density at high frequencies at a relatively small computational cost. On the other hand, the overall approach still had the accuracy problems identified in the original algorithm due to the node correlations created by the reconvergent fanout, and its applicability remained limited to combinational circuits.

Ghosh, *et al.* proposed methods for estimating the average switching activity in combinational and sequential logic circuits [5]. Their methodology was capable of partially accounting for temporal correlations in internal and output nodes, spatial correlations created by reconvergent fanout, and the glitch activity created by spurious transitions.

The primary inputs (PIs) to a circuit were assumed to be independent, and their behavior had to be specified in terms of their transition probabilities. These probabilities for a given signal x were described in the form p_x^{00}, p_x^{01}, p_x^{10}, and p_x^{11} where the superscript ij denoted the probability of x experiencing a transition from state i to j. At the PIs, these probabilities could be obtained from the long term behavior of the input signals. For example, the probability of signal x changing from 1 to 0 could be obtained as $p_x^{10} = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} x(k) \cdot \overline{x(k+1)}$. The signal probability of a node x was obtained from its transition probabilities as $P_s(x) = p_x^1 = p_x^{10} + p_x^{11}$. Consequently, $p_x^0 = 1 - p_x^1$.

The signal probabilities were propagated from the inputs to internal and output nodes using a disjoint cover of the Boolean function of the node in terms of the PIs. Thus, if the disjoint cover of the Boolean function of node y were $f_y(X) = \sum_{k=1}^{m} C_k$, where $X = x_1, \cdots, x_n$ represents the vector of the PIs and C_k is a cube of $f_y(X)$, then $P_s(f_y) = \sum_{k=1}^{m} P_s(C_k)$. Since C_k is the product of independent signals, then $P_s(C_k)$ is the product of the probabilities of the variables in it. Transition probabilities were propagated from the inputs to internal and output nodes by finding the signal probability of the XOR of the Boolean function of each node in two consecutive time frames. Thus for node y, $P_t(y) = P_s(y(t) \oplus y(t+1))$. The resulting expression for $P_t(y)$ can be expressed in terms of the signal and transition probabilities of the PIs.

As an example consider a two-input AND gate, $y = x_1 \cdot x_2$:

$$P_t(y) = P_s[x_1(t) \cdot x_2(t) \oplus x_1(t+1) \cdot x_2(t+1)] = P_s[g(y)]. \tag{2.12}$$

A disjoint cover of $g(y)$ defined as above is

$$\begin{aligned} g(y) &= x_1(t) \cdot \overline{x_1(t+1)} \cdot x_2(t) + x_1(t) \cdot x_1(t+1) \cdot x_2(t) \cdot \overline{x_2(t+1)} \\ &\quad + \overline{x_1(t)} \cdot x_1(t+1) \cdot x_2(t) + x_1(t) \cdot x_1(t+1) \cdot \overline{x_2(t)} \cdot x_2(t+1). \end{aligned} \tag{2.13}$$

Thus, the transition probability of y can be written as

$$P_s\left[g\left(y\right)\right] = p_{x_2}^{1}\left(p_{x_1}^{10} + p_{x_1}^{01}\right) + p_{x_1}^{11}\left(p_{x_2}^{10} + p_{x_2}^{01}\right). \tag{2.14}$$

The implementation of the propagation algorithms for signal and transition probabilities was based on ordered BDDs (OBDDs). To account for gate delay effects in the computation of transition probabilities at a given node y, l Boolean functions of y were constructed, where l was the number of levels found when advancing in the circuit from the PIs to node y. Each of the l functions described the logic expression of y at the time points where it can experience a change. The transition probability of y was obtained as the signal probability of the XOR of each two successive functions of y.

As an example, consider the circuit of Figure 2.3. Assuming the AND gate and the inverter have the same delay, node z could experience two transitions as a consequence of a single change in the input vector. In this case $y\left(1\right) = \overline{x_1\left(1\right)}$, $y\left(2\right) = \overline{x_1\left(2\right)}$, $z\left(1\right) = \overline{x_1\left(1\right)} \cdot x_2\left(1\right)$, $z\left(2\right) = \overline{x_1\left(1\right)} \cdot x_2\left(2\right)$, and $z\left(3\right) = \overline{x_1\left(2\right)} \cdot x_2\left(2\right)$. Thus $P_t\left(z\right) = P_s\left[z\left(1\right) \oplus z\left(2\right)\right] + P_s\left[z\left(2\right) \oplus z\left(3\right)\right]$.

For extending the above technique to sequential circuits, Ghosh *et al.*, proposed a decomposition of the structure of the FSM such that the temporal correlation between two consecutive vectors in the present state (PS) lines could be captured. The transformation started from the common model of a sequential circuit, illustrated in Figure 2.4. By replicating the symbolic simulation equations corresponding to the internal and output nodes in

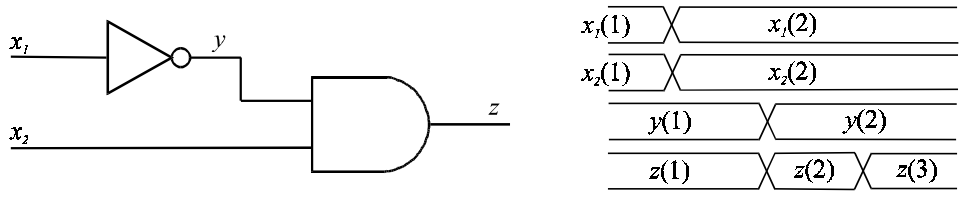


Figure 2.3. Glitch activity caused by gate delays.

the next state logic, using two time instances of the primary inputs, $PI(t)$ and $PI(t+1)$, and using the present and next state lines, two time spaced expressions of the logic functions at the combinational logic block, $f(t)$ and $f(t+1)$ were obtained. Based on this transformation, illustrated in Figure 2.5, the transition probabilities could be obtained as the signal probability of the expression resulting from the XOR of $f(t)$ and $f(t+1)$. The application of this method assumed that the present state lines were spatially uncorrelated and that after power-up, the probability of the FSM of being in any of its 2^N states, where N is the number of state variables, is uniformly distributed. Although the proposed decomposition was very clever, the last two assumptions do not necessarily hold for a large range of sequential circuits, resulting in many cases in considerable estimation errors. The results presented from the application of this approach did not provide a reference measurement which could allow for comparison of the estimation errors.

The technique proposed in [5] to estimate the switching activity in sequential circuits was revisited by Monteiro, *et al.* and Tsui, *et al.* in two simultaneously published studies proposing exact and approximated estimation methods [30, 3]. The exact method proposed by Monteiro, *et al.* used the same decomposition of the FSM structure suggested by Ghosh, *et al.* to capture the temporal correlation in the present state lines. However, instead of assuming uniform state probabilities, the actual steady-state probabilities of the FSM were found when the number of transitions approached infinity ($t \rightarrow \infty$). This computation was performed by regarding the state transition graph (STG) of a FSM as a Markov chain and using the Chapman-Kolmogorov equations to compute its limiting probabilities [21]. The probabilities computed in this way would include the spatial correlations of the present state lines and for this reason it was considered an exact method. However, this approach required an explicit enumeration of the state space of the FSM, implying that the linear system to be solved had an exponential complexity in N, the number of flip-flops of the FSM. To make the problem tractable for large FSMs, an approximate method was proposed which required the solution of a system of nonlinear equations of size N, formulated as follows.

Let $ns_1, \ldots, ns_N$ be the next state lines in a FSM. Then $ns_i = f_i(X, PS)$, where $X =$

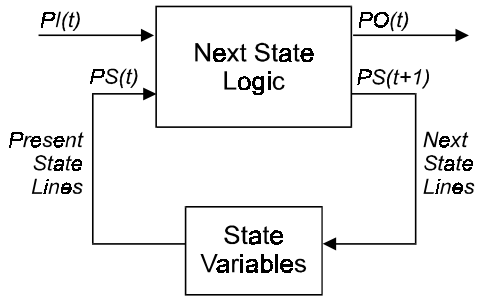


Figure 2.4. Conventional FSM.

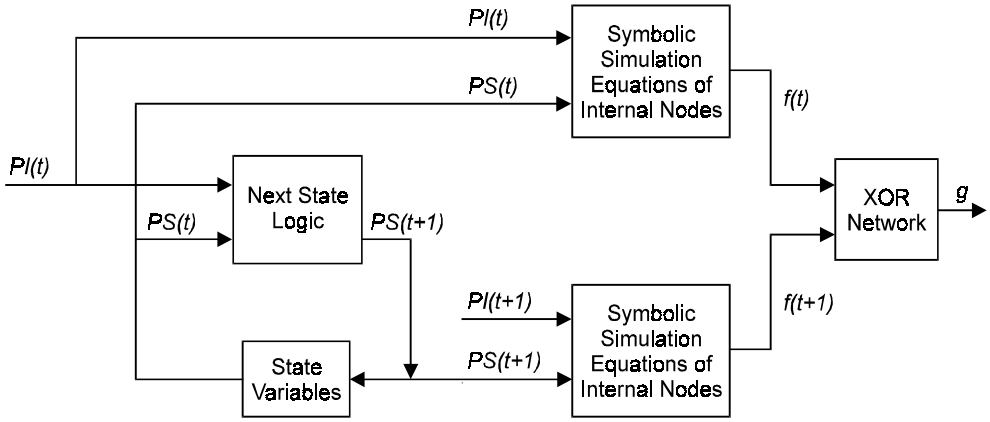


Figure 2.5. FSM transformation to capture temporal correlations.

$x_1, \ldots, x_M$ is the set of primary inputs and $PS = ps_1, \ldots, ps_N$ is the set of present state lines. When the FSM is in equilibrium, $P_s(ps_i) = P_s(ns_i) = p_i$, and therefore $p_i = P_s[f_i(X, PS)]$. Given that the probabilities of the PIs are known, the state line probabilities can be written as $p_i = g_i(p_1, \ldots, p_N)$, where $g_i(\cdot)$ is the function resulting after replacing the values of $P_s(X)$ in $P_s[f_i(\cdot)]$. This yields a set of nonlinear equations in terms of the state line probabilities of the form

$$\left.\begin{array}{c} y_1 = p_1 - g_1(p_1, \ldots, p_n) = 0 \\ y_2 = p_2 - g_2(p_1, \ldots, p_n) = 0 \\ \vdots \\ y_N = p_N - g_N(p_1, \ldots, p_n) = 0 \end{array}\right\} Y(P) = 0. \tag{2.15}$$

The set of equations $Y(P) = 0$ were iteratively solved using the Newton-Raphson method: $P^{k+1} = P^k - J\left(P^k\right)^{-1} Y\left(P^k\right)$, $k = 0, 1, \ldots$ with $P^0 = [0.5, \ldots, 0.5]$. The convergence of the method was proved under the assumption that $\left\|y\left(P^0\right)\right\|$ is small. In each iteration, to evaluate the Jacobian $J\left(P^k\right)$, each entry in the matrix would be $\left.\frac{\partial y_i}{\partial p_j}\right|_{P^k}$. It was noted that for $i \neq j$, $\frac{\partial y_i}{\partial p_j} = -\frac{\partial g_i}{\partial p_j}$ and $\frac{\partial y_i}{\partial p_i} = 1 - \frac{\partial g_i}{\partial p_i}$ making it possible to evaluate any entry in $J(\cdot)$ from $\frac{\partial g_i}{\partial p_j}$. The expression for $\frac{\partial g_i}{\partial p_j}$ could be obtained from $f_i(\cdot)$ through the application of Shanon's expansion theorem as follows: $f_i(X, PS) = ps_j \cdot f_i(X, PS)|_{ps_j=1} + \overline{ps_j} \cdot f_i(X, PS)|_{ps_j=0}$. Taking probabilities and considering that $P_s(ps_j) \to p_j$ as $t \to \infty$ and that the primary inputs were independent, it resulted in

$$P_s\left[f_i(\cdot)|_{P_s(X)}\right] = g_i = p_j \cdot P_s\left[f_i(\cdot)|_{\substack{P_s(X),\\ ps_j=1}}\right] + (1 - p_j) \cdot P_s\left[f_i(\cdot)|_{\substack{P_s(X),\\ ps_j=0}}\right]. \tag{2.16}$$

Then, partial derivatives were obtained as

$$\frac{\partial g_i}{\partial p_j} = P_s\left[f_i(\cdot)|_{\substack{P_s(X),\\ ps_j=1}}\right] - P_s\left[f_i(\cdot)|_{\substack{P_s(X),\\ ps_j=0}}\right], \tag{2.17}$$

where the probabilities in the expression were evaluated using OBDDs. Although this method allowed an accurate computation of the line probabilities, the state probabilities

would not be accurately computed due to the spatial correlations existing among the state lines. For approximating these correlations Monteiro, *et al.* proposed obtaining m-wise correlated sets of state lines, with $1 \leq m \leq N$ a user specified value, and using them to construct the set of nonlinear equations. The accuracy of this approach as well as the number of equations in the system to be solved increased with m, degenerating into the exact method (Chapman-Kolmogorov) when $m = N$. The case $m = 1$ corresponded to assuming spatially uncorrelated state lines. The results of applying this method, when compared to the exact solution obtained through the Chapman-Kolmogorov method, indicated that setting $m = 1$ produced errors of less than 10% on the selected test cases. Using $m = 2$, the maximum recorded error was less than 5% and $m = 3$ produced errors equal or less than 3.6%.

The method proposed by Tsui, *et al.* made use of Ghosh's decomposition of FSMs to capture the temporal correlations in the present state lines [3]. Their exact method also used the Chapman-Kolmogorov equation to compute the limiting probabilities of the Markov chain obtained from the STG of the FSM. Here again, the exponential dependence of the number of equations to be solved on the number of state variables severely limited the size of the largest sequential circuit which could be analyzed with this method. An approximate method was proposed which unrolled and cascaded the next state logic k times to form a k-unrolled network.

The logic equation of the i^{th} bit of the next state outputs at the k^{th} stage were written as

$$\begin{aligned} ns_i^k &= f_i\left(X^k, PS^k\right) = f_i\left(X^k, NS^{k-1}\right) \\ &= f_i\left[X^k, f_1\left(X^{k-1}, PS^{k-1}\right), \cdots, f_N\left(X^{k-1}, PS^{k-1}\right)\right] \\ &\vdots \\ &= f_i\left[X^k, f_1\left(X^{k-1}, f_1\left(X^{k-2}, \cdots, f_1\left(X^0, PS^0\right)\cdots\right)\right), \cdots, f_N\left(\cdot\right)\right]. \end{aligned} \quad (2.18)$$

Taking probabilities, replacing the values of $P_s\left(X^k\right)$, and considering that in equilib-

rium $P_s(ns_i) = P_s(ps_i) = p_i$, a set of nonlinear equations was obtained in the form $p_i = g_i(p_1, \ldots, p_N)$, which could be written as $Y(P) = 0$ with $y_i = p_i - g_i(p_1, \ldots, p_N) = 0$. A solution of this set of nonlinear equations was obtained by using the Picard-Peano iteration method. For a nonlinear function $f(x)$, this method starts with an initial guess x_0 and calculates $x_{i+1} = f(x_i)$ until convergence is reached. A proof of convergence of this method was included under the assumption that $f(x)$ was contractive. The results of applying the above approximation approach indicated that using $k = 3$, the average error relative to the exact method was around 5%.

Tsui, *et al.* later revisited their approximated approach proposed in [3] to compare it to that of Monteiro, *et al.* and found that the Picard-Peano method usually required more iterations but less time than the Newton-Raphson method [31]. This is because Newton-Raphson converges at a quadratic rate while Picard-Peano does so linearly, but the computation of the Jacobian in the Newton-Raphson method is more computationally expensive than the Picard-Peano iteration. A problem found in the Picard-Peano method is that for circuits where the contractiveness condition is not met, the iterations might become oscillatory keeping the method from reaching convergence.

All the probabilistic methods discussed above assumed the primary circuit inputs as independent. This assumption is generally true for data lines, however it does not necessarily hold for control inputs. In typical controller applications, the input patterns are restricted to sequences of commands which are fed to the circuit in some determined fashion. As a consequence, the inputs of those circuits have a pseudo random behavior which typically presents high levels of spatial and temporal correlations. Assuming that the inputs of such circuits are independent frequently leads to considerable errors in the estimates of their switching activities. Accounting for all kind of correlations is a very difficult problem however. The estimation techniques discussed below propose approximate models which account for part of these correlations.

Schneider, *et al.* proposed an estimation technique capable of taking into account concurrent transitions and temporal correlations on the primary inputs of combinational circuits

under a zero-delay model [32]. This technique was proposed as part of a power optimization mechanism which retargeted the functional decomposition of Boolean expressions to find solutions with low power consumption. To determine the transition probability of an internal node in a circuit, a transition function g was built by performing the XOR of two consecutive time instances of the Boolean function of the node expressed in terms of the circuit primary inputs. The signal probability of the transition function was equivalent to the transition probability of the node. Thus, for a given node $y = f(X)$, where $X = x_1, \ldots, x_n$ is the set of primary inputs, $g(y) = f\left(X^0\right) \oplus f\left(X^T\right)$. To evaluate the probability of the transition function, the Boolean expression was recursively decomposed using Shanon's theorem until it was expressed in terms of the probabilities of the primary inputs, which were considered to be spatially independent.

Applying Shanon's decomposition with respect to x_i^0 and x_i^T on $g(y)$ gave

$$g(y) = x_i^0 x_i^T \cdot g(y)_{x_i^0 x_i^T} + x_i^0 \overline{x}_i^T \cdot g(y)_{x_i^0 \overline{x}_i^T} + \overline{x}_i^0 x_i^T \cdot g(y)_{\overline{x}_i^0 x_i^T} + \overline{x}_i^0 \overline{x}_i^T \cdot g(y)_{\overline{x}_i^0 \overline{x}_i^T}. \quad (2.19)$$

Taking probabilities and expressing in terms of the transition probabilities of the primary inputs yield

$$\begin{aligned} P_s[g(y)] \;=\; & P_t(y) = p_{11}(x_i) \cdot P_s\left(g(y)_{x_i^0 x_i^T}\right) + p_{10}(x_i) \cdot P_s\left(g(y)_{x_i^0 \overline{x}_i^T}\right) \\ & + p_{01}(x_i) \cdot P_s\left(g(y)_{\overline{x}_i^0 x_i^T}\right) + p_{00}(x_i) \cdot P_s\left(g(y)_{\overline{x}_i^0 \overline{x}_i^T}\right), \end{aligned} \quad (2.20)$$

where $p_{jk}(x_i)$ represents the probability of signal x_i changing from state j to state k in two consecutive clock cycles. The values of these probabilities are obtained in the primary inputs as $p_{10}(x_i) = p_{01}(x_i) = \frac{1}{2} P_t(x_i)$, $p_{11}(x_i) = P_s(x_i) - \frac{1}{2} P_t(x_i)$, and $p_{00}(x_i) = 1 - P_s(x_i) - \frac{1}{2} P_t(x_i)$.

The implementation of the algorithm was based on the construction of reduced OBDDs (ROBDDs) for each node. A single ROBDD traversal provided the transition probability of the node, which meant that the computational time was linear in the size of the BDD. Note however, that these are global BDDs (in terms of the circuit PIs), which will limit the

size of the largest circuit that can be handled due to their exponential size on the number of PIs. The study did not provide comparative results of the power estimation mechanism.

In another attempt to account for input correlations in probabilistic estimation, Marculescu, *et al.* proposed an analytical model which accounted for spatiotemporal correlations in combinational circuits fed with pseudorandom and biased input streams [33]. Their methodology used lag-one Markov chains to compute signal correlation (SC) and transition correlation (TC) coefficients between pairs of signals in a circuit. Higher order correlations (those which involve three or more signals) were approximated by pairwise correlation coefficients. Once the pairwise correlation coefficients of the input nodes were obtained, OBDDs were used to propagate them through the circuit. Two propagation approaches were evaluated: *global* and *incremental*. In the global approach, the OBDDs of internal and output nodes were built in terms of the PIs of the circuit, while in the incremental approach the OBDDs were developed in terms of the immediate fan-ins of the nodes. The computation of the transition probabilities at an internal node f with support set $x_1, \ldots, x_n$ was carried out using the sets of OBDD paths in the ON-set and OFF-set of f, identified as Π_1 and Π_0 respectively.

The probability of f making a transition from state i to j, denoted $p\left(f_{i\rightarrow j}\right)$; $i, j \in \{0, 1\}$ is the probability of the input vector changing from $\pi \in \Pi_i$ to $\pi' \in \Pi_j$. This probability can be found in terms of the transition probabilities of the PIs and their pairwise transition correlation coefficients as

$$p\left(f_{i\rightarrow j}\right) = \sum_{\pi\in\Pi_i} \sum_{\pi'\in\Pi_j} \prod_{k=1}^{n} \left[p\left(x_{k_{i_k\rightarrow j_k}}\right) \cdot \prod_{1\le k<l\le n} TC^{x_k x_l}_{i_k i_l, j_k j_l} \right]. \tag{2.21}$$

In the incremental approach, the correlation coefficients must also be propagated. If any of the signals in f depend on some other signal x, the transition correlation between f and x can be expressed as

$$TC^{fx}_{i_p, j_q} = \frac{\sum_{\pi\in\Pi_i} \sum_{\pi'\in\Pi_j} \prod_{k=1}^{n} \left[TC^{x_k x}_{i_k p, j_k q} \cdot p\left(x_{k_{i_k\rightarrow j_k}}\right) \prod_{1\le k<l\le n} TC^{x_k x_l}_{i_k i_l, j_k j_l} \right]}{p\left(f_{i\rightarrow j}\right)}. \tag{2.22}$$

In order to reduce the complexity of the operations involved in the evaluation of equations (2.21) and (2.22), a recursive application of Shanon's decomposition was made on f which dropped the evaluation time from complexity $O\left(P^2\right)$ to $O\left(n^2NP\right)$, where P represents the number of paths in the OBDD, N the number of vertices of the graph, and n the number of variables. The results of using both the global and incremental approaches were compared to those coming from logic simulations and a probabilistic approach which assumed independent PIs, on a single test circuit. It was found that ignoring the signal correlations in the PIs under a global approach produced errors as large as 19.5% compared to 0.78% when spatiotemporal correlations were considered. Using the incremental approach, the error dropped from a maximum of 22.6% in the case of uncorrelated inputs to 16.2% in the case of inputs with statiotemporal correlations. The execution time of the spatiotemporal case was 605.1 times longer than that of uncorrelated inputs when using the global approach, and 103.7 times longer in the incremental case.

The problem of estimating switching activity in circuits operating under highly correlated input streams is revisited by Marculescu, *et al.* to include a faster mechanism to propagate transition probabilities and correlation coefficients based on the concepts of *conditional independence* and *isotropism* of signals [34]. Two Boolean functions f and g are conditionally independent with respect to a common variable x if $P_s\left(fg|x\right) = P_s\left(f|x\right) \cdot P_s\left(g|x\right)$. The signals in a set $X = \{x_1, \ldots, x_n\}$ are isotropic if they are conditionally independent for every signal $x_i \in X$, $i = 1, \ldots, n$. These two properties have been asserted as being capable of yielding efficient computational algorithms for the propagation of probabilistic parameters. However, determining the set of signals leading to them has been found to be an NP-hard problem. Marculescu, *et al.* proposed tractable heuristics for establishing approximations of these properties. Sets of signals satisfying the approximation conditions were called *almost conditional independent* (**a.ci**) and *almost isotropic* (**a.is**). Given a set of signals $X = \{x_1, \ldots, x_n\}$ and a signal $x_i \in X$, the subset $X' = \{x_j : 1 \leq j \leq n,\, j \neq i\}$ is

said to be **a.ci** with respect to x_i if there exists an ε, $0 \leq \varepsilon < 1$ such that

$$\left| \frac{\prod\limits_{j \in X'} P_s\left(x_j \,|\, x_i\right)}{P_s\left(\prod\limits_{j \in X'} x_j \,|\, x_i\right)} - 1 \right| \leq \varepsilon. \tag{2.23}$$

If the **a.ci** property holds with relative error ε for every signal $x_i \in X$, $i = 1, \ldots, n$, then X is considered to be **a.is**. For such a set X the signal probability of the product of all its elements can be approximated with error ε by

$$P_s\left(\prod_{j=1}^{n} x_j\right) \simeq \frac{\left[\prod\limits_{1 \leq i < j \leq n} P_s\left(x_i x_j\right)\right]^{\frac{2}{n}}}{\left[\prod\limits_{j=1}^{n} P_s\left(x_i\right)\right]^{\frac{n-2}{2}}}. \tag{2.24}$$

This approximation allowed for redefinition of the algorithm proposed in [33] to propagate the signal and transition probabilities, and the correlation coefficients in the global and incremental approaches. The resulting propagation formulas for a function f whose support set is almost isotropic were

$$p\left(f = i\right) = \sum_{\pi \in \Pi_i} \left(\prod_{1 \leq k < l \leq n} SC_{i_k i_l}^{x_k x_l} \right)^{\frac{2}{n}} \prod_{k=1}^{n} p\left(x_k = i_k\right) \tag{2.25}$$

and

$$p\left(f_{i \to j}\right) = \sum_{\pi \in \Pi_i} \sum_{\pi' \in \Pi_j} \left(\prod_{1 \leq k < l \leq n} TC_{i_k i_l, j_k j_l}^{x_k x_l} \right)^{\frac{2}{n}} \prod_{k=1}^{n} p\left(x_k = i_k\right). \tag{2.26}$$

The definition of SC, TC, π, and Π as well as the propagation formula for the TC coefficients were the same as in [33].

Another modification introduced in [34] was that after some level l in the circuit, the signals were considered uncorrelated. This assumption made it unnecessary to propagate the correlation coefficients beyond that level, reducing the execution time of the algorithm. To validate this modification, a single test case was selected. The newly proposed algorithm was

run on it, and its results were compared to those obtained by assuming independent inputs and those from assuming spatiotemporal correlations with unlimited coefficient propagations. The reference model for error computations was exhaustive simulation. It was found that the additional error introduced by the model assuming spatiotemporal correlations with limited coefficient propagation with respect to the model with unlimited propagations was negligible, but produced speedups between 3 and 30, increasing for larger circuits. When low correlated signals were fed to the test circuit, both the model with uncorrelated inputs and that with spatiotemporal correlations with limited propagations produced estimates within a 10% error. For highly correlated PIs, the uncorrelated model produced 80% of the estimates with errors larger than 10%, while the model with spatiotemporal dependences with limited propagation kept all estimates with errors within 10%. An additional comparison was performed with a method that considers only temporal correlations in the PIs. In the results from this last method, 80% of the estimates had errors of less than 10%. In general, the average speedup achieved by the model considering spatiotemporal correlations with limited coefficient propagation with respect to the original method of [33] was 7.06. This means that still considering uncorrelated inputs is about 15 times faster than the model with spatiotemporal correlations with limited propagations.

Cheng, *et al.* proposed a hybrid methodology for power estimation in digital CMOS circuits merging simulation and probability based techniques [35]. Their approach separated the circuit primary inputs (PI) into control- and data-path lines. This division was based on the argument that control lines show strong temporal and spatial correlations and are typically limited to certain sequences of patterns, while typical data lines are weakly correlated and random in nature. As a consequence, their approach characterized control lines through logic (deterministic) simulation, while data lines were characterized through a probabilistic approach. Both the simulation-based and probability treatment of the circuits was based on the approach proposed by Ghosh *et al.* [5], where transition probabilities are obtained from the symbolic analysis of the Boolean function of each node in two consecutive time frames.

In order properly mix simulation and probabilities, the two time instances of each node were represented by BDDs, where the control inputs were the top variables. Using the supplied input control sequences, a deterministic traversal of both BDDs was performed until the first data input was found. At this point, an XOR of the resulting sub-BDDs was performed and the method continued by symbolically simulating the resulting BDD until the probabilities of the node were found.

A consequence of the simulation-based approach followed for the control inputs is that in the worst case, the above procedure has to be performed on each node a number of times equal to the number of entries in the supplied control sequence. For large circuits, this operation might be too time consuming. For the data inputs it might not be possible to build their complete sub-BDDs. For these cases, a heuristic was proposed which built the sub-BDDs in terms of the internal nodes rather than the PI's, using the concept of *supergates* [36]. The supergate of a node x is the minimal subcircuit in the transitive fan-in of x such that the fan-ins of the subcircuit are logically independent. For large supergates, the heuristic proposes reducing their sizes by considering as independent reconvergent signals arriving to the output of the supergates through paths of maximal distance larger than a user-definable threshold k. As a result, large supergates can be broken down into smaller ones at the expense of some loss of accuracy in the computation of the probabilities. The smaller the value of k, the faster the computation and also the larger the error introduced in the result.

The whole estimation method can also be applied to sequential circuits following the approach proposed by Monteiro, *et al.* in [30] to compute the state line probabilities. The results from the hybrid method were compared to those obtained from using a purely probabilistic approach [5], showing that the former increases its accuracy for circuits with long control sequences. Although no comparison of the execution time was presented, it can be inferred that the hybrid approach should take longer than a purely probabilistic method, especially for circuits with long control sequences which require a large number of steps for completing their logic simulation.

2.2.3 Comparative Analysis and Selection of Estimator

A comparison of the reviewed statistical and probabilistic estimation methods indicates that in general, statistical methods are more accurate than their counterparts. This accuracy advantage comes from the fact that statistical mechanisms are capable of accounting for all kind of correlations in both input and internal nodes. In addition, since the core of these methods is either a logic or circuit simulation engine, delay models other than zero can be accommodated in the estimation mechanism such that the results can include the transitions created by the glitch activity.

Despite all these good characteristics, statistical techniques still have some problems which affect their applicability. First is the problem with the mechanism used to define the stopping criteria for the simulations. Monte Carlo approaches assume normality in the distribution of the activity of the circuit nodes. However many circuits have been found which do not satisfy the normality assumption, leading to considerable estimation errors. Although some nonparametric techniques have been proposed, their models become complex and compromise important characteristics found in Monte Carlo estimation to get rid of the normality assumption. Examples of such characteristics are the non-independence requirement in the primary inputs and the elimination of the capability of providing estimates for internal nodes in reasonable time.

Another problem associated with statistical estimation comes from its application to sequential circuits. Before starting the collection of samples, statistical methods need to ensure that internal activities have reached stationarity, a condition achieved by waiting for a period identified as the *initial transient*. Failure to comply with this requirement leads to biased sampling and consequently improper estimations. Although this a relatively simple process in combinational circuits, it is not clear how to properly determine the length of the initial transient in finite state machines. Chou and Roy [37] have studied this problem and identified the necessity of a solid formulation of a mechanism to determine a proper transient period. They proposed a mechanism for attaining such an objective. However, their formulation requires that a transition matrix Q resulting from converting the expanded

STG of the FSM into a Markov chain be irreducible, aperiodic, and diagonalizable with all eigenvalues greater than or equal to 1 [38]. This last requirement implies that all eigenvalues of Q must be distinct, a condition which combined with the assumption that the STG has to have exactly two near-closed classes, limits the applicability of the procedure to a small range of sequential circuits.

Probabilistic methods, on the other hand, are in general faster than statistical techniques since the former are capable of compactly describing complex input sequences in a few probabilistic parameters. In addition, the propagation of input parameters to internal and output nodes in most cases can be achieved in a single pass of the model representing the nodes in the circuit. The major problem leading to inaccuracies in probabilistic methods is the handling of signal correlations in the propagation mechanisms. Early probabilistic methods were very fast because they ignored all correlations in input and internal nodes. However they incurred considerable estimation errors even when the input signals were truly independent.

Newer techniques considering the dependencies created by reconvergent signals were capable of significantly reducing those errors. Some of these newer techniques have even been able to integrate the effects of glitches in their estimation, increasing their accuracy but reducing their speed. The most recent studies have shown that when circuit inputs are highly correlated it is not enough to consider only the internal correlations. Instead, models must be developed which consider signal dependencies from inputs to outputs. Unfortunately, accounting for all kinds of correlations is a very difficult problem. Several approaches have been reported which introduce approximate models to account for correlations in input and internal nodes in combinational circuits. These studies have achieved increased accuracy at the expense of very complex models and long execution times.

Additional comparative details about statistical and probabilistic estimation methods are summarized in Table 2.1, which lists the features of the representative techniques discussed in previous sections.

The selection of a switching activity estimator for the placement of low power circuits

Table 2.1. Summary of reviewed estimation techniques.

Method	Class	Circ. Type	Primary Inputs	Internal Nodes	Delay Model	Average Speed	Detail of Estimate
Burch [4]	Stat.	Comb.	N.R.	N.R.	Real	Med.	Global
Xakellis [22]	Stat.	Comb.	N.R.	N.R.	Real	Slow	Nodes
Najm [2]	Stat.	Seq.	N.R.	N.R.	Zero	Slow	Nodes
Stamoulis [23]	Stat.	Seq.	N.R.	N.R.	Zero	Med.	Nodes
Yuan [24]	Stat.	Comb.	F.I.	N.R.	n/a	n/a	Global
Cirit [27]	Prob.	Comb.	F.I.	F.I.	Zero	Fast	Nodes
Najm [29]	Prob.	Comb.	S.I.	S.I.	Real	Fast	Nodes
Ghosh [5]	Prob.	Cb/Sq	F.I.	N.R.	Z/U/R	Fast	Nodes
Monteiro [30]	Prob.	Seq.	F.I.	N.R.	Z/U/R	Fast	Nodes
Tsui [3]	Prob.	Seq.	F.I.	N.R.	Zero	Fast	Nodes
Schneider [32]	Prob.	Comb.	S.I.	N.R.	Zero	n/a	Nodes
Marculescu [34]	Prob.	Comb.	N.R.	N.R.	Zero	Med.	Nodes
Cheng [35]	Hybrid	Cb/Sq	FI/NR	N.R.	Zero	Med.	Nodes

must be guided by a reasonable balance between accuracy and speed. The analysis presented above along with the in-depth study of several representative techniques indicate that the selection should be made from the group of probabilistic techniques because of their speed advantage. Within this group, it is necessary to select distinct tools for combinational and sequential estimation. It has been concluded that the techniques proposed by Ghosh, *et al.* [5] for combinational circuits and by Monteiro, *et al.* [30] for sequential circuits produce estimates with a proper speed-accuracy balance, and also provide the option of using either zero- or nonzero-delay models. Although these techniques do not provide the level of accuracy attained by those considering input dependencies, the target application will be minimally affected by this fact. A close analysis of the results reported by Marculescu, *et al.* in [33] indicates that estimates made considering statiotemporal dependencies are correlated to those obtained using independent PIs. In other words, the relationship between high and low active nodes is maintained regardless of the correlation model used in the PIs. Therefore, the relative weights of nodes, measured by switching activity, remains virtually unaffected.

2.3 Power Minimization Techniques

A number of optimization strategies have been reported targeting low power consumption circuits at almost all levels of the design hierarchy. Most of these efforts have been directed at reducing the dynamic component of the power dissipation, given its dominance in the total circuit dissipation. As was mentioned before, the parameters addressed in most power reduction techniques are the supply voltage V_{dd}, the load capacitance C_{Lk}, and the switching activity α_k. A review of approaches affecting these parameters follows.

The dependence of the dynamic component P_{dl} on the square of the supply voltage makes this last parameter a powerful means of reducing the power dissipation. However, the gains attainable by scaling down the supply voltage are limited by the larger delays introduced in the circuit. This fact makes it necessary to introduce some additional mechanisms in the circuit in order to maintain its performance.

Considering the high levels of integration that can be achieved with current CMOS technologies, a feasible approach is to trade power for area. A reduced power dissipation is obtained by scaling down the supply voltage to levels close to those of thresholds, and using hardware replication, parallelism, and pipelining techniques to compensate for the resulting losses of speed. Chandrakasan, *et al.* presents a detailed study of this approach, evaluating the minimum value of the supply voltage which optimizes the balance between power dissipation and hardware overhead [39]. This approach, although reasonable, is highly dependent on the level of exploitable parallelism existing in the circuits, and might not be appropriate for area critical applications which could not afford the trade.

Nielsen, *et al.* has taken as a base the results of some researchers which point out that in certain data-dominated applications, self-timed circuits turn unused parts into a stand-by mode achieving very low power consumption [9]. In their approach, power savings are pushed further by combining self-timed circuits with an adaptive mechanism which adjusts the supply voltage to the smallest possible, while maintaining the performance requirements. This technique, although promising, is impractical for most applications due to the large area overhead introduced in the circuit by the adaptive supply scaling circuitry.

The reduction of the load capacitance and especially the switching activity, has been targeted by different techniques at different levels of abstraction. At the circuit level, transistor scaling and reordering techniques have been developed which target both objectives: capacitance and switching activity [40], [10]. At the logic level, techniques are differentiated by their target circuits, *i.e.* whether they are aimed at combinational or sequential circuits.

In combinational circuits, several techniques have been developed to manipulate logic equations to reduce power dissipation using technology independent optimizations. Examples of these approaches can be found in [7] and [41], where power cost functions of circuits are developed and then minimized using algebraic techniques. Some other techniques have concentrated on manipulating the controllability of the "don't care" set of combinational circuits, in order to reduce the nodal transition probabilities and thus reduce the switching activity [40].

Sobelman and Raatz proposed a delayed evaluation methodology applied to array multipliers to reduce spurious transitions [42]. By using dynamic CMOS circuits, a self-timed precharge/evaluate signal is generated so that each row of adders in the multiplier is evaluated only after all its inputs have stabilized to their final values. In this way, a large number of intermediate transitions are avoided, reducing the switching activity in the resulting circuit, and therefore its power dissipation. Other approaches aim to reduce the glitch activity in combinational circuits by working on the delays of the paths that converge a common signal to a given gate [40]. Approaches aiming at this objective are known as *path balancing* techniques.

At the technology mapping level, some techniques have been developed to include power dissipation as part of the mapping constraints. This task is undertaken by Tiwari, *et al.*, in [12] by developing power models for library gates and applying a technology mapping algorithm which seeks a minimum in power.

The way in which number representations affect the switching activity has been studied by Chandrakasan, *et al.* as a mean of reducing the power dissipation [43]. Their study found that the transition probabilities in data buses could be significantly reduced

in systems devoted to DSP applications by using sign-magnitude representation instead of two's complement. The area overhead introduced by the increased complexity of addition/subtraction operations using sign-magnitude representation is viewed as a trade for reduced power consumption.

In the area of sequential circuits, optimizations in power have been achieved by methods which operate either at the state transition graph level or at the gate and flip-flop levels. State encoding schemes have been modified to target power cost functions weighted by switching activity [44]. Another approach to state assignment in FSMs proposed near-optimal state assignment for minimizing state transitions through the usage of genetic local search algorithms [45]. Here, state machines with less than fifty states were generated with power savings of about 20% with respect to area only optimizations. Moreover, modest reductions in area were also achieved. Schemes dealing with the flip-flop level have also been reported which reposition flip-flops in synchronous sequential circuits to minimize the switching activity [40]. This is a power-oriented application of the technique known as *retiming* [46].

Other techniques have been aimed at eliminating switching activity from idle circuit areas during certain clock cycles. These techniques include the *gated clock* and *precomputation* approaches [8, 47]. The principle of the gated clock technique consists of identifying inaction of particular registers or units and gating their clocks such that they can be selectively disabled. The same approach can be extended to combinational blocks like arithmetic units, in order to turn them off by gating their power lines.

The precomputation technique is similar to clock gating in the sense that it also aims at disabling idle portions of a circuit in selected clock cycles. But in this case the detection of inactive sub-circuits is dependent on the data pattern being processed. This principle is successfully applied in [47], where the logic value of pre-selected units is computed one cycle in advance. Based on this precomputation, decisions are made with respect to which sub-components can be turned off in the next clock cycle. Their results report savings up to 75% in average circuit switching activity, with marginal increases in circuit area and delay.

Power saving techniques have also been applied at the architectural and behavioral levels. These techniques usually require paired estimation tools capable of operating at the same level of abstraction. A methodology and tool developed by San Martin and Knight, aims at minimizing power consumption of circuits during their behavioral synthesis [8]. By combining clock gating, supply voltage scaling, mixed voltage circuits, and trading area for power savings, their tool is capable of mapping behavioral descriptions on profiled circuit modules, under power and/or delay constraints.

Another behavioral synthesis method targeting low power consumption of data-dominated CMOS circuits is presented by Raghunathan and Jha [48]. This approach is specifically aimed at reducing the load capacitance during the allocation phase of synthesis by minimizing the number of functional modules, registers, and multiplexers allocated to the circuit. In order to simultaneously reduce the switching activity, the effect of the behavioral synthesis on signal transitions in the critical data paths is studied by means of functional simulations. The transition information extracted from this step is then used to obtain a near optimal allocation of modules constrained by transition activity. The results of this technique show power reductions up to 33% in certain applications.

A logic synthesis method for combinational logic circuits using Shannon graphs is presented by Lavagno, *et al.* [49]. This method aims at minimizing power consumption due to transition activity under a bounded fanout circuit model. Results indicate power savings greater than 50% for selected circuits, at the expense of increased area. Another technique which optimizes clock tree design for low power by means of inserting buffers in the clock distribution lines has been proposed [50, 51]. The effect of this approach is to partition the clock tree into smaller sub-trees which require smaller wire widths, thus reducing the wiring capacitance. In addition, the buffer size is also optimized under the constraint of a tolerable clock skew. Results in both papers claim power reductions in the clock net of about 50%, while at the same time reducing transition times and buffer area requirements.

As can be appreciated from the above review, most power minimization techniques have overlooked the effect of the physical design and more specifically the effect of the layout

placement on the power dissipation of VLSI circuits. Only two publications, reviewed below, were found to deal with the subject of power reduction at the physical design level.

Vaishnav and Pedram have proposed a performance-driven algorithm for minimizing the power consumption of VLSI circuits during the placement phase [14]. Their technique aims to reduce the weighted net lengths, where net weights are calculated as the expected switching activities of the gates driving the nets. This approach, which targets the placement of cell-based circuits, makes an analytic formulation of the placement problem through a quadratic objective function of the power consumption on each net. This formulation, although capable of producing a global optimal solution, does not accurately model wire lengths, which are the basis for estimating the power consumption of the nets. Their quadratic approach is also combined with iterative partitioning of the circuit until block sizes are reduced to be less than ten gates. At this point a linear assignment is performed. Their results did not report on the time requirements of their placement algorithm and the savings obtained were 7% in power reduction at a cost of 8% increase in wire length and 2% increase in delay.

Chao and Wong discussed in [13] an algorithm for low-power layout design during the floorplan stage. Their technique is designed to take into account the effects of voltage reduction techniques and power-down schemes in the power/ground distribution network, and the thermal distribution in the resulting floorplan design. The basis of their strategy is that the geometry of flexible modules affects their power dissipation patterns. Therefore, given a circuit with p modules, where each module m_i has n_i different geometric implementations, the floorplan cost function can be modified to minimize the power dissipation of the resulting layout within given boundaries of area and delay. This algorithm, although addressing the issue of low power dissipation at the physical level, neglects the power consumed in wires. It mainly concentrates on optimizing the geometry of flexible modules for reducing power and balancing the thermal distribution in floorplans.

2.4 Placement Techniques

Placement can be described as the process of finding suitable physical locations for the components of a circuit on a layout surface, such that some central goal is optimized subject to a set of pre-established design constraints. The placement process typically follows partitioning in the circuit design flow. In the cases where modules have flexible shapes and floating pins, placement is performed at the end of the floorplanning stage. Figure 2.6 illustrates the relative position of the placement process within a typical circuit design flow. The input to the placement phase includes a set of modules, each with a set of associated terminals and a netlist describing the interconnections among modules. As a result of the placement process, a set of coordinates is produced which specify the layout location of each module.

The main objective of placement is to provide a completely routable solution such that the next design stage, routing, can be successfully completed. The space of routable solutions is, in most cases, large enough to allow for the establishment of other objectives which can be concurrently pursued. Typical goals include minimizing the routing area, minimizing the total area, maximizing the circuit performance, or minimizing the routing congestion. Constraints are defined which further reduce the search space. A constraint imposed on virtually any placement mechanism is the avoidance of module overlaps in the output solution. Other constraints imposed include the confinement of the solution to specified bounds and limits in one or more parameters from the list of goals above, other than that selected as the central objective. Objectives and constraints are combined into an objective function constructed to reflect through its value the quality of a placement solution. To assess this quantity, the cost of the central objective is evaluated and penalties are assigned to constraint violations.

Finding a placement solution, regardless of the formulated objective function, has been proven to be an *NP*-complete problem [52, 53]. Therefore, the mechanisms used for finding solutions are, in general, of a heuristic nature.

Placement heuristics can be classified into two broad categories: *constructive* and *iter-*

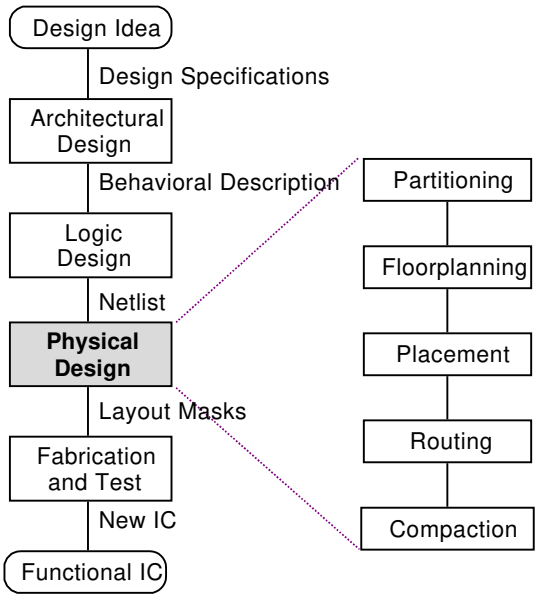


Figure 2.6. Design flow of digital integrated circuits.

ative improvement. Constructive techniques build a placement from scratch, selecting at each step the best module to be placed given the set of previously selected modules. The selection of the first element is usually based on a linear ordering method [54]. Constructive techniques are commonly fast, greedy approaches with time complexity $O(p)$, where p represents the number of modules being placed. Despite their speed, most constructive mechanisms get stuck at local minimum points. Techniques like numerical optimization, placement by partitioning, and some variants of force-directed placement are good examples of constructive heuristics.

Iterative improvement techniques start with an initial placement solution and repeatedly modify it in search of a cost reduction until no significant further improvement is detected. The initial placement is generated by some quick heuristic, typically a constructive technique. Iterative improvement algorithms usually produce good placements, but require considerable amounts of computation time. Examples of iterative improvement are simulated annealing, some force-directed placement techniques, and genetic algorithms.

Among iterative improvement techniques, a further classification is usually made based

on the way new moves are generated. Heuristics that function on the basis of fixed rules are called *deterministic*, and always produce the same result for a particular placement problem. Other techniques work by randomly examining new configurations and may produce a different result each time they are presented with the same problem. These types of heuristic algorithms are called *stochastic* or probabilistic [55].

The scope of application of placement heuristics is related to the layout style of the target circuit. Layout styles usually fall into one of two broad categories: full-custom or semi-custom layout [54]. Full-custom designs are usually hand-crafted by artwork designers using layout editors. As a result, their area and performance are typically at the high end, but at a high cost in man-hours and long time-to-market periods.

In semi-custom designs, some restrictions are imposed on the design elements in order to reduce the complexity of the layout. This simplification in physical design requirements has allowed the automation of a large number of tasks in the layout process, with a consequent reduction in cost and development time.

Depending on how the restrictions are imposed on the circuit layout, several semi-custom design styles can be further defined. The three most common design styles and supporting semi-custom integration technologies are *macro blocks*, *standard cells*, and *gate arrays*. The elements of a macro block-based layout are characterized by arbitrarily sized modules which can be placed virtually anywhere within the layout boundaries. In the standard cell style, all blocks are of same height but with variable widths and can only be placed in predefined rows. Gate arrays are characterized by prefabricated, regular, two-dimensional arrays of modules, with limited routing resources available to provide interconnections. These traditional semi-custom styles can be combined to produce mixed layouts, the combination of macro blocks and standard cells being one of the most common [56], [57]. Figure 2.7 illustrates some of the layout styles cited above.

From the semi-custom technologies described above, the macro block style has the closest resemblance to the full custom layout style. It is capable of achieving very efficient circuit layouts. Large complex logic functions such as datapaths, arithmetic logic units,

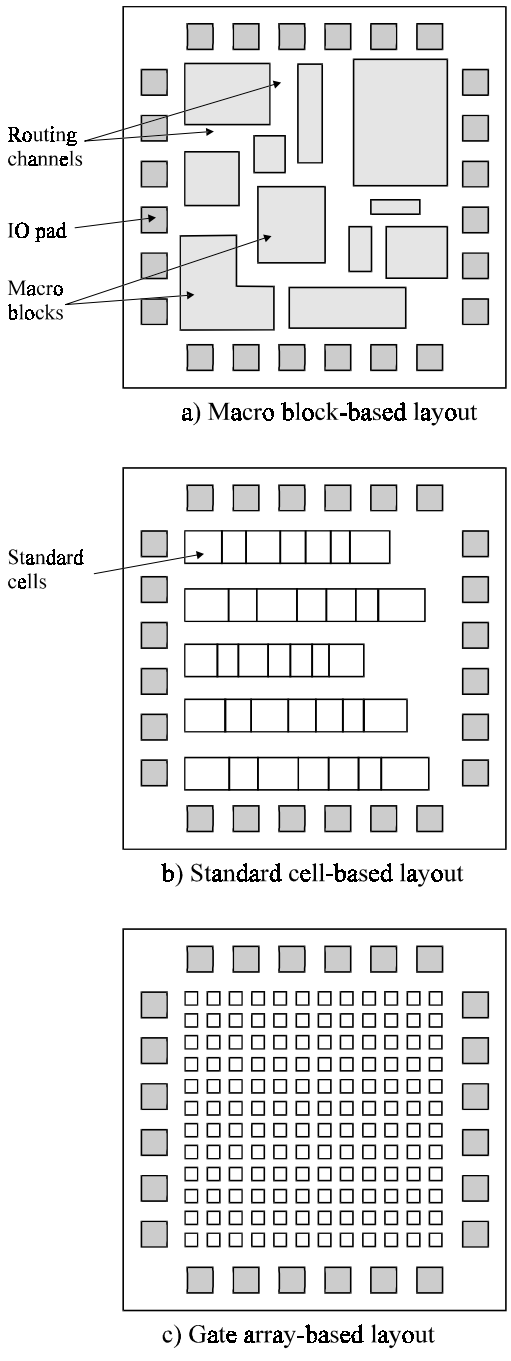


Figure 2.7. Basic layout styles.

and memories use this style. It provides the highest freedom in the placement of circuits and, by introducing the appropriate constraints, the placement of standard cells, gate arrays, and even mixed layouts can be treated as special cases. However, these advantages come at the price of increased complexity in the physical design phase, especially in the stages of placement and routing.

Regardless of the underlying heuristic or target layout style, evaluating the cost of a placement involves measuring the wirelength requirements of its networks. The next section discusses the issues involved in such a measurement and several available techniques.

2.4.1 Wirelength Estimation

Any objective function formulated for the placement problem in any heuristic must consider a cost measurement of the wirelength required by the solution. The actual wirelength of a placement is not known until a detailed routing of the layout is performed. However, performing a detailed routing to determine this quantity each time a placement solution is evaluated is impractical due to the complexity of routing problem itself. Thus, estimators are used in order obtain a representative measure of the minimum wirelength.

The accuracy and speed of the wirelength estimator selected for a placement heuristic has a great impact on the overall quality and time requirements of the heuristic. These factors become especially critical in heuristics using an iterative improvement mechanism. General requirements call for a quick estimator providing results with minimum *skew* error. This last requirement states that the estimation error remain the same for all nets regardless of the number of terminals it contains.

Most wirelength estimators limit routing to *Manhattan geometry*, *i.e.*, routing tracks are either horizontal or vertical. Several alternatives have been reported for wirelength estimation in multi-terminal nets based on a Manhattan geometry. A selected group is discussed below.

Semi-perimeter Method

The most widely used estimator in layout placement is the semi-perimeter method [54]. It involves finding the smallest bounding rectangle that encloses all pins in the net whose length is being estimated. The wirelength is obtained as half the perimeter of the bounding rectangle. The attractiveness of this method is its simplicity, which translates into a fast and easy to implement mechanism. This estimator produces results very appropriate for unobstructed layouts with two- or three-terminal nets. However, in layouts with obstacles and nets with more than three terminals, the semi-perimeter method will always underestimate the wirelength.

Minimum Spanning Tree (MST)

In a minimum spanning tree-based method, the wirelength is estimated as the cost of a minimum length tree spanning all terminals of the net. Exact algorithms with polynomial time complexity have been developed for this task [58]. Estimators based on MSTs can provide good estimates in nets with four or less terminals. Beyond this limit, it has been found that the estimation error, although bounded, gets worse as the number of terminals per net increases [59]. In many cases, MSTs are used as approximations of Steiner trees (discussed below). An important finding relating these two mechanisms is that the worst case performance ratio of the cost of an MST to the cost of a Steiner minimum tree is bounded by $\frac{3}{2}$ [60].

Steiner Minimum Trees

A Steiner minimum tree (SMT) provides the shortest route to interconnect a multi-point net. However, determining optimum Steiner trees has been proven to be an *NP*-complete problem [61]. The minimum Steiner tree problem on a Manhattan plane, formally known as the Minimum Rectilinear Steiner Tree (MRST) problem, is frequently defined as follows: given a set P of n points in a plane, find a set Z of additional points (Steiner points) such that the MST of $P \cup Z$ has minimum cost, the cost of any edge in the tree is the rectilinear

distance between its end points, and the cost of the tree is the sum of the costs of its edges [62].

Finding computationally efficient solutions to the MRST problem has been an extensively researched area, and still is, due to its relevance in VLSI design and other areas. Significant simplifications are introduced into the problem by the Manhattan geometry of circuit layouts, restricting the problem to rectilinear dimensions. An early finding related to this restriction is that the set of candidate Steiner points can be reduced to those falling on the intersections of escape segments determined by block boundaries and terminal points [63]. Further, clever techniques for identifying and removing certain kinds of redundant or unfeasible candidate points and wire segments have been developed, especially for the obstacle free case [64].

Despite all these reductions in the solution space, the MRST problem has remained *NP*-complete [65]. A few techniques have been reported which can find exact solutions of small problem instances [66]. Most efforts have been directed at developing time efficient heuristics to find approximate solutions, frequently aimed at unobstructed layouts [67–69]. A relatively small number of techniques are aimed at layouts with obstacles [70]. A common characteristic of most of these techniques is that their approximations are based either on an iterative improvement of a minimum spanning tree, or on partitioning the problem into small instances which are optimally solved and then joined together by shortest paths.

Another class of approaches has undertaken the SMT problem by introducing "Steinerizing" modifications to traditional algorithms for finding shortest paths or MSTs and applying them in a greedy fashion. This class of heuristics, commonly applied to the Steiner problem in networks, has been reported to have polynomial time complexity and a bounded worst case performance [71–74]. These properties among others, are highly desirable in an SMT heuristic selected for wirelength estimation in a low-power placement application. The reasons for this statement are discussed later in Chapter 4.

Other Estimators

A number of other methods have been used as wirelength estimators, which have found limited applicability mainly because of accuracy problems. These include the *complete graph*, *minimum chain*, and *source-to-sink connection* methods.

The complete graph method approximates the wirelength of a net N with n terminals to the average length of a tree spanning all terminals in the net. The complete graph of N has $\frac{n(n-1)}{2}$ edges, while a tree of the same net has $(n-1)$ edges. Then, the average tree length of N can be obtained as $\frac{2}{n}\sum(\textit{length of all pairs})$. This computation takes time $O\left(n^2\right)$ and tends to overestimate the net length.

The minimum chain method assumes that all terminals form a chain, making each node have *degree* two. The chain is constructed starting from an arbitrary terminal and continues adding the next closest terminal at each step, without breaking the degree rule, until all terminals are in the chain. This technique also tends to overestimate wirelength.

In the source to sink connection method, a star is constructed centered at the net driver node, extending a branch to each input terminal. This estimator performs best in heavily congested layouts, but yields excessive interconnect length in those with average to low congestion.

Figure 2.8 shows an example layout illustrating the operation of the wirelength estimation mechanisms discussed above.

2.4.2 Partition-based Techniques

In a partition-based technique, a layout is repeatedly partitioned into two sub-circuits. At each step, the *cutset*, *i.e.*, the number of nets connecting both parts, is minimized, subject to some balance condition on the *cutsize*, *i.e.*, the individual areas of the two partitions. This process is carried out until each subcircuit can accommodate only one cell. These methods are based on either the Kernighan-Lin or the Fiduccia-Mattheyses partitioning heuristics [54]. One of the earliest applications of partitioning for placement is due to Breuer [75]. He proposed several variants of the technique based on changes to the objective function and

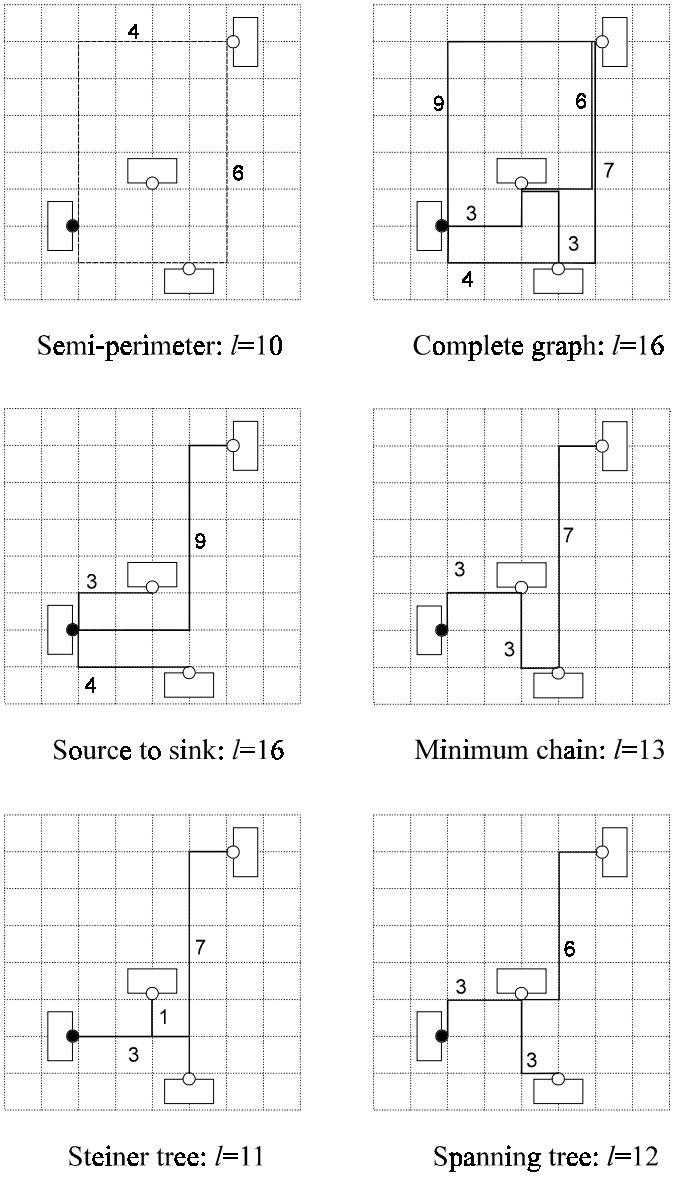


Figure 2.8. Wirelength measures produced using different estimation methods.

on the sequence in which the cut lines are selected. An extension of this basic concept was made by Lauther for macro-block based layouts by introducing an improvement phase in which module rotation and mirroring operations are allowed [76].

One of the major drawbacks of these earlier techniques was the exclusion in the objective function of the effect of external terminal positions in the partitions, resulting in many cases in unnecessarily long interconnects. An improvement introduced by Dunlop and Kernighan called the *terminal propagation method* was aimed at solving this problem [77]. This method introduces dummy cells in the subcircuit to be partitioned next whenever a net is to be cut. This forces the partitioning algorithm to assign adjacent sections to subcircuits sharing the same nets, avoiding positions for the subcircuits which would otherwise require longer wiring.

Another modification in partitioning techniques was introduced by Suaris and Kedem in a method called *quadrisection* [78]. This technique extends terminal propagation to a mechanism which partitions the layout horizontally and vertically in a single step. The argument for this method is that it achieves a true two-dimensional placement, rather than the one-dimensional orientation of the bisection method. This requires cell exchanges between four quadrants rather than two while keeping the same objective of minimizing the cutset.

The main advantage of partitioning techniques, in general, is that considering subcircuits as semi-independent problems considerably reduces the size of the search space at each successive step. This, however does not guarantee that the obtained solution will be a global optimum because the partitioning problem itself is *NP*-complete. Even in the cases where an optimum partition can be found at each step, the optimum placement is not guaranteed because block exchanges among partitions do not have a global effect on the layout.

2.4.3 Force-directed Methods

This class of placement techniques establishes an analogy between layouts and mechanical systems by comparing modules and their interconnecting networks to a system of masses

joined by springs. The analogy establishes that blocks interconnected by nets exert attractive forces on each other. The magnitude of the force between any two modules is directly proportional to the distance separating them and the constant of proportionality is the relative weight of the nets. Thus, the application of Hooke's law dictates that if modules were allowed to move freely, they would move in the direction of the force until the system achieves equilibrium in a minimum energy state where the resultant force on each module would be zero. This would be equivalent to the minimum wirelength of the interconnecting nets among modules.

Many variants of force-directed placement techniques have been reported, some are constructive heuristics that use numerical optimization techniques, while others are based on iterative improvement. The group of constructive heuristics work by setting-up a system of equations of the forces exerted on each module. Equating the system to zero and simultaneously solving, it is possible to find a solution for the placement of each module [79], [80]. There is no guarantee that the solution produced will be the global minimum because the resulting system of equations will be nonlinear.

Iterative improvement methods begin with an initial solution from which one module is selected at a time and its zero force position found. The selected module is then either placed on the target location (*force-directed relaxation*) or exchanged with the module previously occupying the target position (*force-directed pairwise relaxation*) [81], [82]. It has been found that this type of force-directed heuristic tends to place all modules in the center of the layout, mainly due to the inherent embedding of the center of gravity formula in the method [83].

2.4.4 Genetic Placement Techniques

A genetic algorithm (GA) is a search technique modeled on the mechanics of natural selection. Candidate solutions are modeled as *individuals*, groups of individuals form *populations*, and successive populations are called *generations*. Each individual in a population is represented by a single *chromosome*, which is a string of symbols called the *genes* of the individual. A set of genes that make up a partial solution is called a *schema* [84].

A genetic algorithm begins with an initial, generally random, population. During each generation the individuals in the current population are evaluated using some measure of *fitness*. The fitness value is used to select a pair of individuals (*parents*) whose genes are applied a number of genetic operators to produce a new solution called an *offspring*. The offsprings are then evaluated and a new generation is formed with selected parents and offsprings according to their fitness, such that the population size remains constant. Since the probability of selecting specific individuals increases with their fitness, the fitness of the entire population improves from one generation to another, improving the overall solution quality. In the application of GAs to the placement problem, single individuals can be formed with a vector of triplets, each triplet representing a module ID, and its plane coordinates in the current solution. The fitness measure could be made to reflect the inverse of the wirelength and module overlap. An initial population would be a number of different random placements.

The most commonly used genetic operators are *crossover*, *mutation*, and *inversion*, which are derived from the biological process of evolutionary genetics. Crossover, the main genetic operator, operates on two individuals at a time to generate an offspring by combining schemata from both parents. This would be equivalent to generating a new placement solution by combining selected sections of the placement vectors of two solutions chosen by their fitness. The mutation operator produces incremental random changes in the offspring produced by crossover. Typical mutations in placement include pairwise exchange, single module displacement, and orientation changes. The proportion of genes mutated in a population (*mutation rate*) has to be carefully chosen so that it produces useful genes without destroying the offsprings' resemblance to their parents. The inversion operator takes a schema from a chromosome and inverts it end for end, to change its representation without modifying the present solution. This mechanism increases the chances of inheriting sections of good genes in future crossover operations. Here again, the rate of inversions per population has to be properly set to control the amount of gene group information in the population. Several approaches for the placement of standard cells using genetic algorithms

have been reported [85], [86]. A few other approaches have targeted this method for the placement of macro blocks [87].

An approach related to genetic algorithms called *simulated evolution* has also been reported for the placement problem [88]. This approach has some basic distinctive features which include the following. The algorithm operates on only one configuration at a time and the modules in the configuration are treated as the population. Generational changes are caused by mutations induced on the individuals. The goodness of the placement of each module is evaluated in terms of its contribution to the total placement cost. Modules with low goodness have a high probability of being tagged during a judgement phase for later reallocation.

In general, genetic techniques are of a stochastic nature and have the capability of producing near-optimal results at the expense of long computational times. Unlike other methods, GAs use information from the history of previous configurations and keep several configurations simultaneously in each iteration, which provides a wider pool of alternatives for generating new configurations. These features however, make GAs one of the methods with the largest memory requirements and highest computational requirements when evaluating new solutions.

2.4.5 Simulated Annealing

The concept of annealing was introduced into combinatorial optimization by Kirkpatrick *et al.*, in the early 1980s [89]. This concept is based on a strong analogy between the physical annealing process of solids and large combinatorial problems. In physics, *annealing* is a two-step thermal process for obtaining low energy states of a solid in a heating bath. In the first step, the *temperature* of the heat bath is increased to the maximum value at which the solid *melts*. During the second step the temperature of the heat bath is carefully decreased until the particles arrange themselves in the ground state of the solid. In this state, the particles in the solid are arranged in a highly structured lattice and the *energy* of the system is minimal. Failure to execute either of these two steps will take the system into a meta-stable state other than the ground state [90].

The core of simulated annealing (SA) is the *Metropolis* algorithm, which simulates the annealing process at a given temperature T_k until the solid achieves thermal equilibrium [91]. The algorithm is based on Monte Carlo techniques and generates a sequence of states of the solid in the following way. Given a current state i of the solid with energy E_i, a subsequent state j with energy E_j is generated by applying a *perturbation* mechanism which transforms the current state into a next state by a small distortion. Denoting the energy difference between states i and j by $\Delta E = E_j - E_i$, state j will be accepted as the next state with a probability P_{ij} given by

$$P_{ij} = \begin{cases} 1 & \text{if } \Delta E < 0 \\ \exp\left(\frac{-\Delta E}{T_k}\right) & \text{otherwise.} \end{cases} \tag{2.27}$$

This acceptance rule, also called the *Metropolis criteria*, defines a hill climbing policy in the exploration of the solution space, which allows the annealing process to escape local minimum points. An implementation of the SA algorithm invokes the Metropolis procedure at various monotonically decreasing temperature values, beginning from an initial temperature T_0 at which the system is said to be molten. If the lowering of the temperature is done appropriately, and a sufficiently large number of states are explored, the system can reach thermal equilibrium at each temperature step. It has been demonstrated that if thermal equilibrium is achieved at each temperature step, the simulated annealing process will have an asymptotic behavior which, if allowed to run for a theoretically infinite time, will converge to the set of globally optimum solutions of the system with probability one [92]. Therefore, simulated annealing is asymptotically an optimization algorithm. In practical applications however, asymptotic convergence can only be approximated, converting SA into an approximation mechanism falling in the category of iterative improvement stochastic heuristics.

The application of simulated annealing to the placement problem establishes the following analogies. The states of the physical system are represented by the different placement configurations that can be given to the layout. The energy of a state is represented by the

cost of its corresponding placement. The temperature is modeled by a control parameter which is reduced according to a schedule designed to allow the achievement of thermal equilibrium at each step. These analogies make SA a conceptually simple heuristic. Algorithm 2.1 outlines an implementation of simulated annealing. The Metropolis procedure, invoked in Step 2.1, is outlined in Algorithm 2.2.

Algorithm 2.1 (Simulated Annealing)

Input: *S_0 = Initial solution, T_0 = Initial temperature*

Output: *Near-optimal placement solution*

1. Initialization
   - 1.1. $T \leftarrow T_0$
   - 1.2. $S \leftarrow S_0$
   - 1.3. $time \leftarrow 0$
2. While (*Stop criterion not satisfied*)
   - 2.1. $S \leftarrow \textbf{Metropolis}(S, T, time)$
   - 2.2. $T \leftarrow \textbf{Decrement}(T)$
3. Return(S)

Algorithm 2.2 (Metropolis)

Input: *S = Current solution, T = Current temperature, time = Current time count*

Output: *Quasi-equilibrium placement solution*

1. While (*Inner-loop criterion not satisfied*)
   - 1.1. $newS \leftarrow \textbf{Perturbate}(S)$
   - 1.2. $\Delta E = \textbf{Cost}(newS)$ – $\textbf{Cost}(S)$
   - 1.3. If ($\Delta E < 0$ or $\textbf{Random}(0,1) < \exp\left(\frac{-\Delta E}{T}\right)$
     - 1.3.1. $S \leftarrow newS$
   - 1.4. $time \leftarrow time + 1$
2. Return(S)

The central tasks in the implementation of a simulated annealing-based placement strategy are the development of the perturbation function, the cost function, and the annealing

schedule. The perturbation function should be designed to introduce small changes to the current solution. This is typically done by performing pair-wise module exchanges and displacing or changing the orientation of single modules. The cost function evaluates each attempted solution according to the objective function defined for the problem. This part of the algorithm has a great impact on the running time of the heuristic due to the amount of time during which it is invoked in the execution. Thus, considering that new states contain only small perturbations of the current solution, it is desired that the cost evaluation be performed in an incremental way to speed up the process. Another consideration of importance in the design of the cost function is the accuracy of the evaluation mechanisms on which it is based. The success of the SA heuristic is tied to accurate estimation of the parameters that define a good solution, especially for those having the greatest influence in the cost function.

The annealing schedule establishes the criteria driving the SA algorithm to produce approximations of a global optimum solution in finite-time implementations. Designing an annealing schedule requires specifying the following four parameters.

- The initial temperature of the process T_0, also called the *melting temperature*: this is a value of the control parameter T which causes acceptance of virtually all proposed transitions.

- The number of iterations performed at each temperature step or *inner-loop criterion*: this criterion is based on the requirement that at each temperature step the system should reach a quasi-equilibrium condition, resembling the thermal equilibrium in a physical system.

- The *temperature decrement function*: this is the rate at which the temperature will be decreased. It has been found that for small decrements of T, the stationary distribution of the state transitions in subsequent temperature values will be close to each other [90]. Therefore, if the system is in quasi-equilibrium at the current step, it is expected that re-establishing it in the next step will take a relatively small number of transitions.

- The final value of the temperature or *stop criterion*: the selection of this parameter is based on an extrapolation of the deviation of expected value of the cost function from a global minimum when the temperature T tends to zero. In practical implementations the criterion is based on the variation of average value of the cost function with respect to the change in temperature. When this ratio is less than some pre-defined small positive number the annealing process is stopped.

The search for adequate annealing schedules has been the subject of extensive research. Experimentally deduced schedules have been reported with good results in some placement applications [93]. Several general schedules, ranging from conceptually simple to relatively complex are discussed by Van Laarhoven and Aarts in [92]. Adaptive schedules have been reported with applications to SA in the placement problem [94], [95], [96]. Most of these approaches use statistical measures to dynamically determine the parameters that define the annealing schedule. This has the advantage of self-adapting the schedule to the problem being optimized, speeding up the convergence of the SA heuristic.

Other approaches have proposed radically different annealing schedules with warming or periodic patterns of temperature changes [97]. A warming schedule begins with a low temperature value that is monotonically increased. A periodic schedule allows the temperature to oscillate between a pair of maximum and minimum values. In both cases, the process continues until a certain expected solution quality is achieved. The final solution produced under such types of schedules corresponds to the best seen during the entire annealing process. Experiments with these techniques reported in certain types of optimization problems point to shorter solution times than those using traditional schedules for the same expected quality of results.

One of the most successful applications of SA to the placement problem has been the TimberWolf package, originally developed by Sechen and Sangiovanni-Vicentelli [98]. This package can accommodate mixed layout styles and performance-driven objectives [99], [100]. Other approaches have reported parallel implementations of SA [101].

In general, simulated annealing has been one of the placement mechanisms most exten-

sively researched and many authors agree in saying that it is one of the most well developed placement methods available. The quality of results obtained with SA are undoubtedly among the best and the technique is conceptually simple to implement. The annealing algorithm keeps track of only one placement solution at a time or two in the case of best-so-far strategies that also keep a record of the best solution visited so far. This gives SA a memory and a speed advantage in the evaluation of new solutions over its closest competitor, genetic algorithms. One of the drawbacks of SA (and also of GAs), is the total running time of the heuristic. The search for mechanisms to make SA to converge faster to a solution has been the force driving all the research in efficient annealing schedules. Among these, statistical annealing approaches have reported good results. Despite these reports, tuning of specific annealing schedules still remains essentially an experimental task.

2.4.6 Other Approaches

A number of other related approaches to the placement problem have been reported. These include techniques using numerical methods, neural networks, and fuzzy decision systems. Numerical methods transform placement into a numerical optimization problem and use some computationally intensive deterministic technique to find a feasible solution. Some of the force directed techniques previously discussed fall into this category. An inconvenience of numerical techniques, in general, is that placement is a nonlinear problem. To deal with this, most numerical methods use either linear approximations or nonlinear programming techniques. Reported approaches include the eigenvalue method also referred to as quadratic assignment, resistive network optimization, the penalty function method (PFM), linear programming and others. A discussion of these methods can be found in [83].

Several approaches have been reported which used neural computing techniques to find solutions to the placement problem. Early studies reported poor solutions and were limited to small problem instances mainly because of the size of the models and their large computational requirements [102]. Later advances have reduced the computational burden of early approaches and reduced their memory requirements, reporting improved results for relatively large problem instances [103], [104]. However, neural techniques in general have

had limited success in solving the placement problem.

Some techniques have been reported using fuzzy decision systems. A constructive method using fuzzy logic was reported by Lin and Shragowitz [105]. In another report, Ball *et al.* proposed a fuzzy mean field approach for the linear and two-dimensional assignment problem [106].

The placement mechanisms discussed in this section have had in general, limited success. Numeric methods, like any deterministic approach are usually fast and greedy, getting stuck at local minimum points. Neural and fuzzy approaches still need more research to determine if they can make an impact in providing solutions to the placement problem and are not mature enough to be considered competitive with simulated annealing.

2.5 Summary

The review presented at the beginning of this chapter exposed the relationships among circuit parameters and input data patterns in the power dissipation of digital circuits, identifying those which can be controlled by power reduction strategies. An analysis of the problem of estimating switching activity in CMOS circuits was presented next. Salient techniques were discussed in detail, highlighting the different estimation trends and identifying their strengths and weaknesses. When considering the applicability of these techniques in a power driven placement environment, it was found that probabilistic models for combinational and sequential circuits, capable of accounting for internal spatial and temporal correlations and independent primary inputs, provided a reasonable speed-accuracy balance in their estimates.

The review of power minimization techniques made evident that although numerous approaches have been proposed with different degrees of success, the physical design stage has received the least attention. Moreover, a review of placement techniques indicates that while this has been the most intensively researched physical design problem, its effects on the power dissipation of digital circuits has been virtually neglected. It becomes clear then that there is a gap in the development of power conscious design techniques for digital circuits

in the physical design area. The objective of the low-power placement tool developed as part of this dissertation is to partially fill this gap. At the same time it aims to create the basis for the development of other physical design tools with the same objective.

The algorithm selected for the placement methodology was simulated annealing and the target style was macro block-based layouts. The review of layout styles and placement techniques presented in this chapter indicates that this selection corresponds to the most general layout style and the most well developed and successful placement method.

CHAPTER 3

Problem Formulation

This chapter has two central objectives: first, the elaboration of the statement of the placement problem for low-power dissipation, and second to provide a global overview of the approach followed in the search of a suitable solution. The first section defines relevant terms and concepts used in the elaboration of the problem statement. The actual statement is presented in Section 3.2. The last two sections describe the approach followed towards a suitable solution and a summary of the ideas discussed in the chapter.

3.1 Terminology and Definitions

In order to provide a formal statement of the placement problem for low power design, this section extends the traditional definition of the placement problem to include the power minimization goal as a dominant component in the objective function of the optimization mechanism. Prior to the statement itself, a few relevant concepts are defined.

Definition 3.1 (Location) *A location is a point in a two-dimensional space. Therefore, a location l_j is specified through an ordered pair (x_j, y_j).*

Definition 3.2 (Module) *A module is a closed area in a two-dimensional space. The portion of the space occupied by a module m_i is called the module's* domain *D_i. A domain D_i is specified through a sequence of points $D_i = \{(x_{i1}, y_{i1}), (x_{i2}, y_{i1}), (x_{i2}, y_{i2}), \cdots, (x_{ic}, y_{ic}), (x_{i1}, y_{ic})\}$. Any two adjacent points in sequence*

D_i have at most three different values and define a boundary *in the module's domain. In D_i, points (x_{i1}, y_{i1}) and (x_{i1}, y_{ic}) are adjacent. An arbitrary point (x, y) is considered to be inside D_i if the sequence specifying the domain describes a counter-clockwise trajectory around (x,y). In a domain sequence D_i, point (x_{i1}, y_{i1}) defines the origin of coordinates for all points in the domain, including those in its boundary. The assignment of location l_j to module m_i, changes D_i by shifting (x_{i1}, y_{i1}) to (x_j, y_j).*

Definition 3.3 (Terminal) *A terminal t is a point on a boundary of a module domain. Terminals provide the gateways to convey* signals *into and/or out from modules.*

Definition 3.4 (Network) *A network, also named* net *or* node, *is a set of terminals which share the same signal.*

3.2 Problem Statement

The problem of placement of macro block cells for low power dissipation can be stated as follows.

Given a set of modules $M = \{m_1, m_2, \ldots, m_p\}$ with terminal sets $T_1, T_2, \ldots, T_p$, respectively, a set of locations $L = \{l_1, l_2, \ldots, l_q\}$, $(q \geq p)$, and a set of networks $\mathbb{N} = \{N_1, N_2, \ldots, N_r\}$ carrying signals $s_1, s_2, \ldots, s_r$, respectively; where these sets satisfy the following relations:

- Each module $m_i \in M$ is associated to a set of networks $N_{m_i} \subseteq \mathbb{N}$, where $N_{m_i} = \{N_j \,|\exists t \text{ such that } t \in N_j \text{ and } t \in T_i \,, 1 \leq j \leq r\}$.
- Each network $N_k \in \mathbb{N}$ is associated to a set of modules $M_{N_k} \subseteq M$, where $M_{N_k} = \left\{m_i \,\middle|\, N_k \in N_{m_j} \,, 1 \leq i \leq p\right\}$.
- Each signal s_k has associated a factor α_k and a function $g_k\,(N_k)$, where α_k is the switching activity of s_k, and $g_k : N_k \rightarrow \Re$ is a function returning the capacitance of the wires interconnecting the terminals that make up network N_k.

Find an assignment S of each module $m_i \in M$ to a unique location $l_j \in L$ such that the cost of the assignment, denoted $C(S)$ and measured by the weighted summation of equation (3.1), is minimized, subject to the condition that the intersection of the domains of any two modules is empty (no two modules can overlap).

$$C(S) = \sum_{k=1}^{r} [\alpha_k \cdot g_k(N_k)] \tag{3.1}$$

Term $C(S)$ in equation (3.1) actually represents the switched capacitance of the network wires in the placement solution S. This value is directly related to changes in the factor $\alpha_k \cdot C_{Lk}$ in component P_{dl} of the total power dissipation defined in equation (2.3), as shown below.

Analyzing in some detail the load capacitance C_{Lk} appearing in equation (2.3), it can be seen that its value can be expressed as the sum of three components, as shown by equation (3.2).

$$C_{Lk} = C_{ok} + C_{wk} + C_{ik} \tag{3.2}$$

In this formula, C_{ok}, C_{ik}, and C_{wk} represent the equivalent capacitances of the net driver output, the net fan-in, and the wires interconnecting the terminals in the net, respectively. In the search for a power efficient solution S, the capacitance component affected by changes in the placement of modules is C_{wk}, since the portion corresponding to $C_{ok}+C_{ik}$ will be the same for any solution. This establishes the indicated direct relationship between equation (3.1) and changes in the P_{dl} component of the total power dissipation. Moreover, in typical macro-block circuits the value of C_{wk} is several orders of magnitude larger than $C_{ok}+C_{ik}$. This implies the relevance of minimizing equation (3.1) in a power optimizing placement technique.

Equation (3.1) can be taken to an even more elemental level. The value of C_{wk} results from the equivalent capacitance between metal and polysilicon layers and the substrate. This capacitance can be approximated using a parallel-plate model similar to that in equa-

tion (3.3).

$$C_{wk} = \frac{\varepsilon}{t} A_k \tag{3.3}$$

In this model, ε represents the permittivity of the insulating material between layers, t is the insulator thickness, and A_k is the plate area. The model of equation (3.3), although not considering fringing fields, multi-layer structures, and mutual capacitance of adjacent wires, provides a reasonable approximation of C_{wk} for the placement application. More accurate models which consider those factors enumerated above and others can be found in [107].

The plate area A_k in equation (3.3) can be obtained as the product $A_k = w_k \cdot d_k$, where w_k and d_k represent the width and length, respectively, of the wires making-up net k. Considering that changes in placement do not affect parameters ε and t, and assuming that $w_k = w$ is constant for all nets, C_{wk} becomes a direct function of d_k. Thus, a reduction of the switched capacitance, and therefore of the power dissipation of the circuit, can be effectively achieved by reducing the product $\alpha_k \cdot d_k$ of the networks of the circuit. If function $g_k(N_k)$ is redefined to return the wirelength of net N_k, the objective function of the placement problem can be re-written as equation(3.4).

$$D(S) = \sum_{k=1}^{r} (\alpha_k \cdot d_k) \tag{3.4}$$

This equivalent objective function, now named $D(S)$, is called the *switched length* of the circuit. Note that wirelength and switched length are in general different objective functions, and in fact, in some cases they can be conflicting objectives, as illustrated in the case below.

Consider the circuit in Figure 3.1, where each gate, considered as a block ($p = 8$), is to be placed in rectangular layout. To reduce the size of the solution space it is assumed that the layout contains a number of locations equal to the block count ($q = 8$), that module dimensions as well as inter-module spaces are of unit length, and that input and output pads are attached to their associated modules. This last assumption reduces the net count to 7 ($r = 7$) with an equal number of signals, whose switching activities have been previously

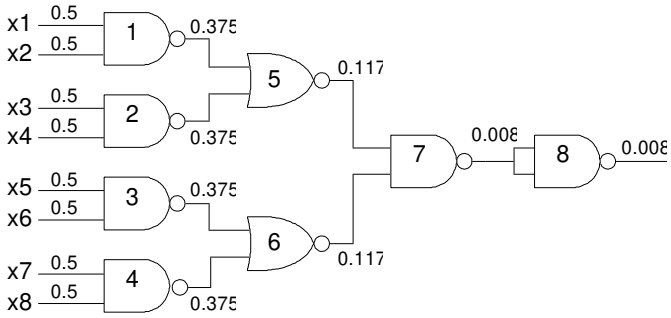


Figure 3.1. Circuit to be placed with estimated switching activities per net.

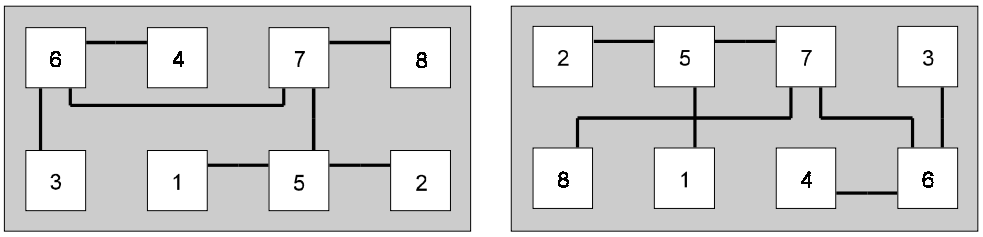
estimated. Despite these simplifying assumptions, the solution space $\mathbb{S}$ remains quite large, $|\mathbb{S}| = 8! = 40320$.

Figure 3.2 shows two solutions for this problem. The first one optimizes wirelength only, while the second optimizes the switched length of the layout. Note that optimizing one parameter renders the other suboptimal. This situation, typical in most circuits, makes it necessary to re-define the objective function guiding the mechanism used to optimize a placement for low power dissipation.

The example discussed above highlights two important characteristics of the placement problem for low-power. First is its intractability. The placement problem has been proved to be *NP*-hard regardless of the optimization objective and the simplifying assumptions introduced by the specifications of the application [52]. Second is the importance of the correct estimation of the switching activity α_k and wirelength d_k of the nets of the circuit.

3.3 Approach

The central objective of this work is to develop a placement methodology capable of producing circuit layouts with near-minimal switched length. As was established in the previous section, this will result in a reduction of the total power dissipation of placed circuits with respect to conventional techniques which traditionally target the expected wirelength for



a) Solution 1: $d_k = 9$, $\alpha_k d_k = 1.975$ b) Solution 2: $d_k = 11$, $\alpha_k d_k = 1.883$

Figure 3.2. Placement solutions with objective functions for: a) wirelength, b) switched length.

reducing either the overall delay or layout area. This new placement methodology also aims at a higher objective of supporting the potential development of complementary tools which, when combined, could provide a complete physical design methodology for low power circuits.

In order to achieve these goals, a bottom-up approach was followed in which each task was specifically designed with the power reduction goal in mind. An outlined description of the most relevant tasks is presented below. The relationships between tasks are graphically depicted in Figure 3.3.

Task 1: *Development of a power annotated netlist format.* Given a circuit specified through a set of design rules, a standard physical netlist, and a functional netlist, the first task consisted of generating a netlist format for optimizing the power dissipation of the circuit through its physical design. This task turned out to be a relevant step in this research, due to the lack of power dissipation parameters in conventional netlist formats. The new format is used to specify both the input and output of the placement process. The transformation process is made with the aid of the switching activity estimator previously selected. The new format is made externally available for use in other design tools that could be developed for reducing the power dissipation of circuits during the physical design process.

Task 2: *Design of localized representations of the layout geometry and network set.* In

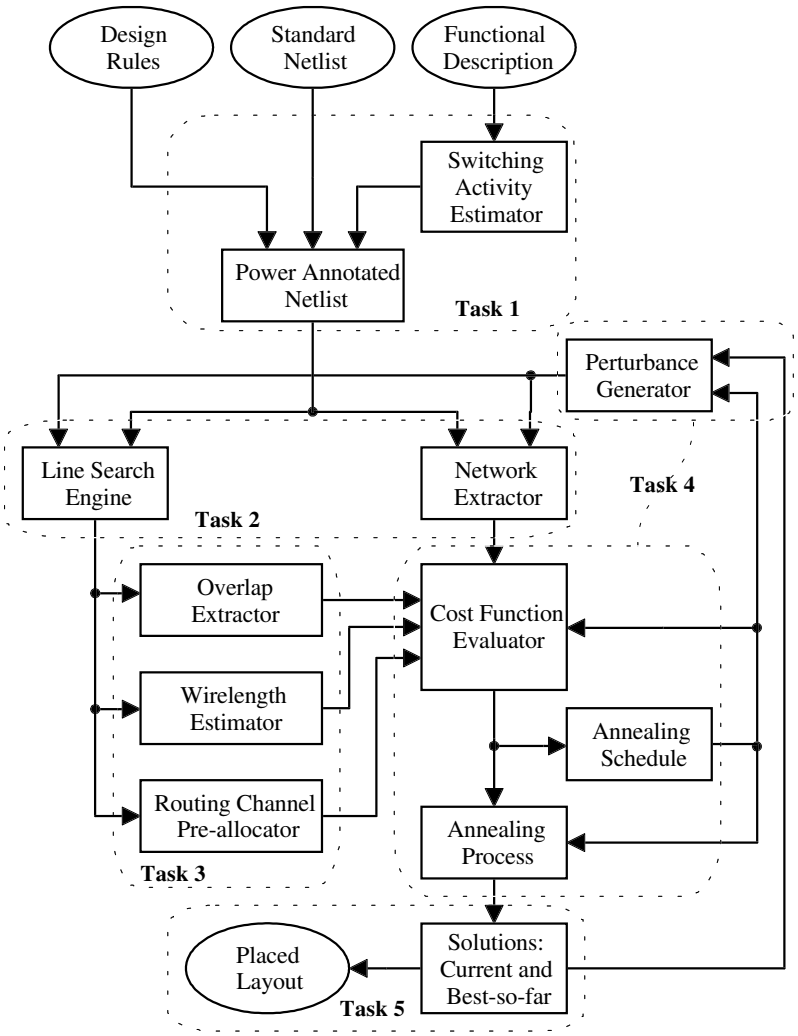


Figure 3.3. Relationships among tasks in the solution approach.

this task, a suitable internal representation of the circuit was designed which allowed an efficient application of algorithms that extract and update placement dependent information. This was one of the key factors for the speed of the optimization mechanism, due to the vast number of placement configurations generated in the processing of a single layout. The designed representation captures the geometric features of the circuit through a line search engine. The same mechanism also provides information to a network generation mechanism which handles the geometrical and electrical characteristics of the network structures.

Task 3: *Design of cost evaluating algorithms.* The efficient computation of the amount of overlap, switched length, and general routability of the circuit at each step of the optimization process were key factors for assessing the accuracy and speed of the evaluation of new placement solutions. In this task, algorithms for computing these parameters were developed. The algorithms were designed to execute on the internal representation of the circuit designed in Task 2.

Task 4: *Implementation of the optimization mechanism.* As previously established, the mechanism selected for optimizing the circuit placement was simulated annealing. In this task, the algorithms that make up the core of the simulated annealing process were designed. These algorithms were designed to adapt themselves to the problem being solved, especially the determination of the annealing schedule and its interaction with the solution perturbation mechanism. Additional algorithms developed in this stage included those dedicated to the collection of statistics for the calibration of the parameters governing the annealing process.

Task 5: *Extraction of data, validation, and evaluation.* In this last task, a number of experiments were performed to collect data from the placement of a set of benchmark circuits selected to ascertain the performance of the newly designed placement tool. Comparative analyses were carried with three central purposes: first, to validate the results against those obtained with an established placement tool. Second, to assess the gains in power reduction achieved with the new tool, and third, to measure the scaled sub-optimality of the heuristic.

3.4 Summary

This chapter provided a statement of the problem of placing a circuit for low-power optimization and reviewed the outlined course of actions executed for providing a solution to it. Terms and definitions essential for the statement were given at first. The statement itself, in addition to presenting the problem with some formality, made evident the complexity involved in the search of a solution. The outlined task description provided a map of the remaining chapters where the detailed descriptions of the procedure are presented.

CHAPTER 4

New Steiner Tree Heuristics for Physical Design Problems

This chapter presents two new time-efficient greedy heuristics for the minimum rectilinear Steiner tree (MRST) problem in graphs. These heuristics, originally developed for the wirelength estimation problem in the low-power placement methodology of this dissertation, can also be applied to diverse problems found in the physical design process of VLSI circuits and in other areas.

The organization given to the chapter begins with a discussion of the reasons why new heuristics were needed for the low-power application. Next, definitions of the terminology used throughout the rest of the chapter are given. The algorithms used to transform circuit layouts into suitable graphs for application of the MRST heuristics are discussed in Section 4.3. This section also describes a set of reduction mechanisms applied to the graph prior to the MRST heuristics to limit the size of the search space. Section 4.4 describes the developed heuristics and their performance analysis. The last two sections present experimental results of the application of the heuristics and a summary of the results of the chapter.

4.1 Why more MRST heuristics?

Finding a minimum length network interconnecting sets of points in the plane is a central task in the physical design of VLSI circuits, particularly at the stages of placement and

routing. Regardless of the objective function used in either or both of these stages, the end results of these procedures will be affected by the quality of the algorithm used to find the interconnecting network. Within a typical placement mechanism, a quick wirelength estimator like the semi-perimeter method is commonly used. In the routing stages, although some other more accurate but slower technique is used, in most cases the behavior of the route estimation heuristic in the largest, least frequent nets is not a main concern. This approach frequently yields practical results in cases where objective functions are not severely affected by non-homogeneous errors in the estimates. Further, the errors from the placement stage have little or no effect in the routing stage, since typical wirelength estimators used in placement do not provide detailed route information required in upper stages.

A different situation arises when optimizing the placement or routing of macro block-based layouts for low-power dissipation. Here the goal is to reduce the power dissipation of a circuit by reducing the wire capacitance of highly active interconnects. In this case, the placement objective function is especially sensitive to the wirelength of specific nets since the capacitance of a net is a direct function of its length. An additional requirement introduced by the layout style is that the interconnecting mechanism has to consider networks with multiple points and obstacles in the routing plane. Moreover, the information obtained from such nets during placement would benefit the global routing phase when providing continuity to the common objective of reducing the switched capacitance.

Under these requirements, a logical approach would be to use an accurate wirelength estimator during placement, capable of yielding results within a reasonable time frame. In addition, if the results contain enough detail, they can be used later to guide the channel definition and assignment process during the global routing phase. Using these guidelines and considering the review of wirelength estimators presented in Chapter 2, it can be concluded that the best estimation mechanism for such a constrained application should be based on a fast and yet accurate SMT heuristic. This makes the class of greedy SMT heuristics referred to in Chapter 2 the first choice in the selection process.

A detailed analysis of several reported heuristics in this class revealed that although their worst case performance is bounded, their estimation error becomes larger with the number of points to be connected by the tree [71–74]. Moreover, it was suspected that larger nets in typical circuits had a significant activity, making them critical nets in the power dissipation of the layout. To verify this suspicion an analysis of switching activities versus net sizes was performed on a group of selected circuits from the ISCAS benchmark suite [108]. A set of four circuits that included sequential and combinational designs was selected providing a total of 1323 nets with terminal counts between 3 and 57 terminals per net. The switching activity estimator previously selected was applied and the data obtained plotted as a scatter of activity versus terminal count per net is shown in Figure 4.1. The figure also shows the central tendency of the data, measured as the average activity per net size. The figure confirms the initial suspicion: an increase in the terminal count per net does not imply a reduction in the switching activity of the nets. In fact, single nets with as many as 56 terminals were found to have activities over 0.30, consuming as much as 6% of the total power in circuits with over 600 nets.

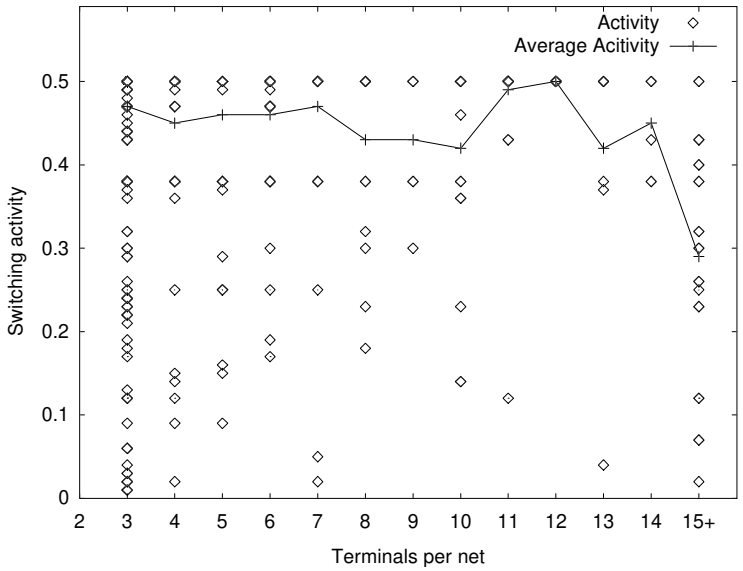


Figure 4.1. Switching activity *vs.* number of terminals in selected circuits.

Therefore, an error skew in the wirelength estimation worsening as the terminal count per net increases is an undesirable characteristic for an estimator used in the low-power placement or routing application. An error skew of this type is, however, unavoidable in a heuristic estimator. Thus, the search was focused on an SMT-based estimator with a better handling of nets with large terminal counts. This search was unsuccessful, thus the development of a new heuristic with the desired characteristics was undertaken.

The remaining sections of this chapter discuss the development of two new time-efficient greedy SMT heuristics which, given a layout graph of a circuit instance containing n vertices from which t ($t \leq n$) are terminals, find an approximate solution in $O(n^2 \cdot \log n)$ time. Both heuristics grow trees guided by the distance of the graph vertices measured with respect to the terminal vertices. In order to produce data sets from actual layouts, the implementation of algorithms for transforming macro block circuit netlists into graphs is discussed. Graph reduction algorithms are developed and implemented to reduce the search space of the SMT heuristics. To measure the quality of solutions and the suitability of the proposed heuristics for the low power placement problem, they are evaluated using a set of benchmark circuits and compared to three other known approaches of their same class. The results obtained highlight the strengths and limitations of each of them.

4.2 Terminology and Definitions

The terminology used throughout this chapter includes definitions found in classical graph theory literature as well as terms which apply to the specifics of the developed algorithms.

Given a layout with obstacles, an **escape segment** is a horizontal or vertical line which extends along the width or height of the layout to either the internal perimeter of the routing region or to the border of an obstacle without intersecting either of them [109]. An **escape grid** is obtained by drawing escape segments around obstacles and through each terminal point along an unobstructed direction [70]. A escape grid can be transformed into a **layout graph** by representing the grid intersections as vertices and the segments between intersections as edges.

Thus, a layout graph $G(V, E)$ is defined as a connected, undirected, edge-weighted graph representing a layout with obstacles. In this notation, V represents the resulting finite set of vertices and E represents the set of unordered pairs of distinct vertices which symbolize the edges. Edge weights are determined by a cost function w such that $w : E \rightarrow \Re^{+}$ through the assignment of the segment lengths as weights. Set V is made up of subsets H and T, where T contains the vertices resulting from grid intersections falling on terminals of the layout. Subset H includes the vertices resulting from the intersections of escape segments at points where there are no terminals. Layout graphs are also referred to as *escape graphs* [70].

Within a layout graph $G(V, E)$, the **distance between two vertices** $d(v_i, v_j)$ is the length of a shortest path from v_i to v_j. If $d(v_i, v_j)$ is maximal with respect to all other pairs of vertices in G, then $d(v_i, v_j)$ is the **diameter** of G, $D(G)$, and v_i and v_j are **peripheral vertices** in the graph. The **distance of a vertex** $d(v)$ in graph G is the sum of the distances from v to all other vertices in G. The **median** $M(G)$ of a graph G is the subgraph induced by the set of vertices having minimum distance [58].

When computing the distance of a vertex u considering only terminal vertices as destinations, the value obtained is the **terminal distance,** $d_t(u)$, given by

$$d_t(u) = \sum_{\forall v_i \in T} d(u, v_i). \tag{4.1}$$

The **terminal diameter** $D_t(G)$ of a layout graph $G(V, E)$ is defined as the length of a longest shortest path between two terminal vertices. The **terminal median** $M_t(G)$ of a layout graph G is the subgraph induced by the set of terminals having minimum terminal distance and the shortest path connecting them. Note that $M_t(G)$ might contain non-terminal vertices as well. A $v_i - v_j$ path is said to be **median biased,** $P^{+}_{v_i v_j}$, if it runs over a set vertices with minimum terminal distance. Note that $d(v_i, v_j)$ will not change if the path on which it is defined is median biased.

4.3 Layout Graph Extraction and Reduction

The concept of using layout graphs for routing multi-terminal nets in the presence of obstacles was studied by Ganley and Cohoon [70]. They proved that if a problem instance is solvable, then there is an optimal solution which is a subgraph of its escape graph.

By using a line sweep algorithm, the generation of all line segments from a problem instance can be accomplished in time $O(m \cdot \log m)$, where m represents the sum of the number of terminal points to be interconnected plus the number of module sides in the layout [109]. Finding the intersections of the line segments can be completed in the worst case in time $O(m^2)$ if all horizontal and vertical segments intersect among themselves [110]. This step produces all vertices and edges of the layout graph. Figure 4.2 shows an example of a macro block-based circuit layout and its corresponding layout graph.

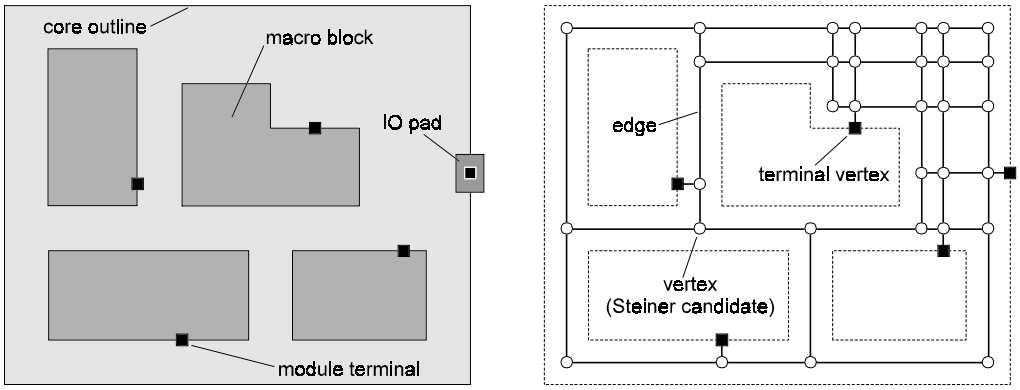


Figure 4.2. Macro block layout and its equivalent graph.

Layout graphs generated in this manner often contain vertices and edges that do not lie in any optimal solution. Reduction algorithms can be used to identify and eliminate these elements from the graph thus reducing the search space for the heuristic used to find a solution. Techniques capable of deleting as much as 90% of the potential Steiner

points created in the graph constructed from unobstructed layouts have been reported [64]. However, not all of these techniques can be directly applied to the kind of graphs generated for this application either because of compatibility or time efficiency reasons. Some of these reductions require transformations which are possible only in graphs obtained from obstacle-free layouts. Additionally, not every technique applicable to graphs obtained from layouts with obstructions is suitable for the target application. Some of them are too computationally expensive. For these reasons, only a selected set of reductions were adapted (when necessary) for use in layouts with obstacles.

The first type of reduction deletes non-terminal vertices of degree two and their incident edges. In the construction of most layout graphs, at least four such vertices will be initially produced in the corners of the layout. These vertices can be easily detected during the construction of the graph. If v_c is a vertex with $\deg(v_c) = 2$, with adjacent vertices v_x and v_y forming a path $P_1 = \{v_x, v_c, v_y\}$, it has been shown that v_c can be deleted from a Hanan grid by *establishing* an alternate $(v_x, \cdots, v_y)$ path P_2 of the same cost as P_1 which does not contain v_c [64]. This reduction can be applied to escape graphs by limiting the selection criteria to searching for the *existence* of a path P_2 such that $Cost(P_2) \leq Cost(P_1)$. Note that this criteria does not require finding a shortest $(v_x, \cdots, v_y)$ path, making it possible to use a breadth-first search (BFS) algorithm to find P_2. Such an algorithm would identify a suitable path in $O(p)$ time, where p is the total number of vertices in the input graph. Moreover, since the graph was directly obtained from the layout geometry, positional information of vertices can be used to reduce the scope of the search algorithm thus reducing its average run time.

The search for potentially removable vertices of degree two can be efficiently performed by holding a FIFO queue initialized with those vertices identified during the graph generation stage as potential candidates. As removable candidates are identified and deleted from the graph, new candidates may be created. These candidates can be added to the FIFO queue for later processing.

The second type of reduction involves those candidates of degree two which do not

satisfy the alternate path condition. In this case, the vertex is removed and its two incident edges are joined into a single edge connecting vertices v_x and v_y.

A third type of reduction involves non-terminal vertices whose degree drops to one as a result of other reductions. These vertices are unconditionally removed from the graph as they are created. Figure 4.3 shows the layout graph previously obtained in Figure 4.2 after the application of the reduction algorithms.

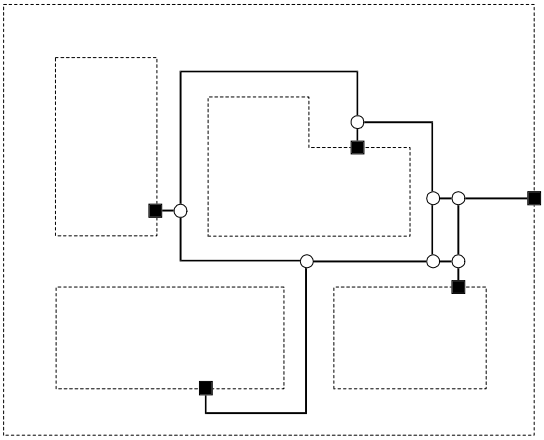


Figure 4.3. Reduced layout graph of Figure 4.2.

This set of reductions can be performed in a single pass over the input graph with $n \leq p$ terminals through the use of the queue indicated above. The worst case time complexity of these computations would be $O\left(p-n\right)^2$ for the case when all non-terminal vertices had to be checked for alternate paths.

4.4 Median Biased Trees

Given a layout graph $G\left(V,E\right)$ with a set of terminal vertices $T \subseteq V$, the problem of estimating the wirelength of the interconnection of the set of points in T can be regarded as that of finding the cost of a Steiner minimal tree (SMT) on a network. This problem calls for

the establishment of a minimum length subnetwork $S^*(T)$, spanning all terminal vertices in the graph. To offer an approximate solution to this problem, two heuristic algorithms which grow Steiner trees guided by the terminal distance of the vertices of the graph are presented. In this way, the solutions they provide will be in the context of *median biased trees*.

The rationale behind growing median biased trees comes from the observation that in many practical cases, an SMT solution runs over a set of vertices with minimum terminal distance. Thus, by biasing the growth of a tree by this metric, the heuristics search for these type of solutions.

The operation of both proposed heuristics is preceded by a pre-processing of the layout graph to identify and eliminate redundancies and infeasibility in shallow vertices. The criteria described in the previous section are used to perform these reductions on the input graph.

4.4.1 Median Biased Heuristic (MBH)

This heuristic starts by labeling all vertices in the reduced graph with their terminal distances. This allows one to identify the terminal median of the graph. A terminal vertex of smallest median distance is selected as root of the tree. If there is more than one, an arbitrary one is selected. The rest of the tree is grown by selecting the remaining terminal vertices of G in non-decreasing order of their median distance and connecting them to the partially built tree through a median biased shortest path. The MBH is outlined below in Heuristic (4.1).

Heuristic 4.1 (MBH) *Finds an SMT approximation on a graph.*

Input: *Reduced layout graph $G(V,E)$ with terminal set $T \subseteq V$, $|T| > 2$.*

Output: *Median Biased Steiner Tree $S(T)$.*

1. Initialization*: The solution tree S is initialized as empty. All vertices are labeled with their terminal distances. As each terminal vertex is labeled, it joins an ascending priority queue indexed by the terminal distance of its elements.*

Let $S \leftarrow \emptyset$. $\forall v_i \in V$, *find* $d_t(v_i)$. *If* $v_i \in T$ *then insert* v_i *into* APQ.

2. Root*: A terminal with smallest terminal distance is removed from the queue to become the root of the solution tree.*

   Remove r *from* APQ *and do* $S \leftarrow r$.

3. Selection*: A terminal* x *with smallest terminal distance is removed from the queue to be added to the solution tree.*

   Remove x *from* APQ.

4. Growth*: A vertex* c *in the solution tree closest to* x *is identified. A median biased shortest path* P_{xc} *is added to the solution tree.*

   Find a $c \in S$ *such that* $d(x,c)$ *is minimum. Establish* P_{xc}^{+} *and add it to* S.

5. Termination*: If all terminals have been processed the solution tree is returned. Otherwise the next terminal is processed.*

   If $APQ = \emptyset$ *then output* S, *otherwise return to Step 3.*

End

To analyze the time complexity of the above algorithm, it will be assumed that $|V| = p$ and $|T| = n$, where $n \leq p$. The initialization step requires finding shortest paths from all vertices to all terminals and sorting them by their terminal distances. By using a heap implementation of Dijkstra's algorithm the distance calculation can be completed in time $O\left(p^2 \cdot \log p\right)$. By also implementing APQ as a heap, the sorting of terminals can be completed in time $O(n \cdot \log n)$, rendering the distance computation as the dominant process in Step 1. The use of heaps allows for Steps 2 and 3 to be completed in $O(\log n)$ time, while the growth step requires $O(p \cdot \log p)$ time. Since Steps 3 through 5 are repeated n times, their complexity will be less or equal to that of Step 1. Therefore, the overall time complexity of the algorithm is $O\left(p^2 \cdot \log p\right)$.

4.4.2 Diameter Biased Heuristic (DBH)

In this heuristic, like in the case of the MBH, the initialization step labels all vertices with their terminal distances. However, the solution tree is not initialized as empty. Instead, two peripheral terminals are identified and used to establish a median biased path whose length is equivalent to the terminal diameter of the reduced layout graph. This path will be the trunk of the solution tree. After this, the remaining terminals are selected one at a time in non-decreasing order of their terminal distances and connected to the partially built tree through a median biased shortest path.

The reason for selecting a median biased terminal diameter as the trunk of the solution tree lies in the observation of MSTs where such a path was frequently part of the solution. In addition, a terminal diameter provides a lower bound in the solution of the MST, as shown in Lemma 4.1.

Lemma 4.1 *Let $D_t(G)$ be the terminal diameter of a layout graph $G(V,E)$ measured between vertices x and y from set T. Then for any SMT $S^*(T)$ of G it is verified that*

$$|S^*(T)| \geq D_t(G). \tag{4.2}$$

Moreover, if $|S^*(T)| = D_t(G)$ then $S^*(T)$ is a median biased shortest $x-y$ path.

Proof: Let $D_t(G)$ be a terminal diameter in a layout graph G measured between terminals x and y. The first part of the Lemma can be proven by contradiction. Assume there exists an SMT S^* in G such that $|S^*| < D_t(G)$. Then, since both x and y are part of S^*, this implies that there exists an $x-y$ path P_{xy} in S^* satisfying $|P_{xy}| < d(x,y)$. Since S^* is a subgraph of G, then P_{xy} can be also be established in G, contradicting that $D_t(G)$ is the length of a shortest $x-y$ path in G. Therefore the relation of Equation (4.2) must be satisfied.

The proof of the second part of the Lemma requires showing that if an SMT S^* of G has length D_t then all terminal vertices fall in a shortest $x-y$ path P_{xy}. Note that if there were at least one vertex $z \in T$ such that $z \notin P_{xy}$, then removing z and all

non-terminals connecting it to P_{xy} would yield $D_t < |S^*|$ contradicting that $|S^*| = D_t$. So, all non-terminal vertices falling in P_{xy} will also be shortest path connected to all terminals in G. This implies that their terminal distances are minimal, making P_{xy} median biased.□

The outline of the DBH, as can be inferred from the above discussion, is very similar to that of the MBH. The main difference lies in Steps 1 and 2, which are re-stated as indicated below:

Heuristic 4.2 (DBH) *Finds an SMT approximation on a graph.*

⋮

1. Initialization*: All vertices are labeled with their terminal distances. As each terminal vertex is labeled, it joins an ascending priority queue indexed by the terminal distances of its elements.*

   $\forall v_i \in V$, *find* $d_t(v_i)$. *If* $v_i \in T$ *then add* v_i *to* APQ.

2. Trunk*: Identify a pair of peripheral terminals* (x, y) *and establish a median-biased shortest path* P_{xy} *between them. Make the solution tree equal to* P_{xy}.

   $S \leftarrow P_{xy}$ *where* P_{xy}*is median biased and* $|P_{xy}| = \max(d_t(u, v)) \forall (u, v) \in T$.

⋮

End

As in the MBH, the time critical part of this heuristic is in Step 1 during the labeling of the vertices of the graph. Therefore its time complexity will remain as $O\left(n^2 \cdot \log n\right)$.

Two additional steps could be added to both MBH and DBH which if inserted after the termination step could, in some cases, improve the quality of the solution without affecting the overall time complexity of the heuristics.

⋮

6. *Re-arrangement*: Build a minimum spanning tree S' for the subnetwork of G induced by the set of vertices in S.

7. *Pruning*: Remove one at a time from S' all non-terminal vertices of degree 1.

Given that Heuristics (4.1) and (4.2) provide approximate solutions, it is of interest to know how far from the optimal these solutions could be. The analysis methodology presented by Takahashi and Matsuyama in [71] can be applied to the proposed heuristics with minor modifications to provide answers to these questions.

This methodology shows that the worst case solution of the class of heuristics to which MBH and DBH belong are upper bounded by twice the optimal solution of the Steiner tree. Specifically, for a problem with $|T| = k$, the relation of the cost of the heuristic solution S to the cost of the optimal solution S^* is given by

$$\frac{|S|}{|S^*|} \leq 2\left(1 - \frac{1}{k}\right). \tag{4.3}$$

The methodology also shows that this bound is tight, *i.e.* it is not possible to obtain a lower upper bound through scaling cases for which the relationship asymptotically tends to the limit [71].

4.5 Experimental Results

To assess the quality of the solutions provided by the MBH and DBH, their performance was compared to those of other heuristics considered as close competitors in terms of the approach used to produce solutions. The heuristics selected for comparison were the median heuristic proposed by Diané and Plesník (DMH) and the distance-network heuristic (DNH) proposed by Kou *et al.* [72], [73].

The DNH operates as follows. Given a graph $G(V,E)$ and a terminal set $T \subseteq V$, begin constructing the distance network of G, $DN(G)$. Next, obtain an MST Q on $DN(G)$ and prune it one vertex at a time, until obtaining a tree Q' where all leaves are terminal vertices. Tree Q' is returned as S. The distance network of G, $DN(G)$ is a complete graph defined on set T such that any edge between terminals v_i and v_j in $DN(G)$ is a shortest $v_i - v_j$ path in G.

The second reference heuristic, the DMH, operates as follows. Given a graph $G(V, E)$ and a terminal set $T \subseteq V$, begin by finding the median of G, $M(G)$. Then, find a set of shortest paths from $M(G)$ to all vertices in T and construct on it an MST K. Prune K to form a tree K' where all leaves are terminals. Next find a DNH solution Q' for $G(V, E)$, T, and return $S = \{D, |D| = \min(|K'|, |Q'|)\}$.

In addition to the DNH and the DMH, the results of the MBH and the DBH were also compared to those of a very simple MST heuristic. In this case, a solution S is constructed by building an MST of the layout graph and pruning all non-terminal vertices, one at a time, until the leaves of the tree are all terminal. This heuristic is based on Prim's algorithm and will be referred as the MST. The reason for including such a simple heuristic is to evaluate if the added time complexity of the other four heuristics is worthwhile in terms of the quality of the results.

A set of five macro block based circuit layouts were selected from the MCNC benchmark suite (`apte`, `xerox`, `hp`, `ami33`, and `ami49`) to support the experiments. The layouts in these data sets contained obstructions whose number varied between 9 and 49 blocks, with about one thousand nets whose number of terminals ranged between 2 and 74. The circuits were given an arbitrary placement before being processed.

The selected layouts and their networks were transformed into appropriate layout graphs over which the SMT-based wirelength estimation heuristics would work. Prior to the application of the heuristics, the reductions described in Section 4.3 were applied on the input graphs. This preprocessing significantly reduced the size of the graphs, limiting the search space of the heuristics. The reduction results are summarized in Table 4.1.

It is observed that the average reduction obtained with the implemented techniques decreased as the number of terminals per net increased. However, it is also noticed that in more than 90% of the nets, the average reductions exceeded 95% of the Steiner candidates in the input graph. Although these results include two-terminal nets, such entries were considered only to report the outcome of reduction techniques. The Steiner estimation results consider only nets with three or more terminals.

Table 4.1. Results of graph reduction algorithms.

Terminals per net	Proportion in data (%)	Average redution (%)
2	73.52	99.58
3	15.54	97.33
4	3.61	96.09
5	1.97	87.14
6	1.31	91.50
7	0.88	81.16
8	0.33	95.27
9	0.88	56.70
10+	1.97	41.18

The reduced layout graphs of each circuit and their nets were fed to the five heuristics under evaluation. To measure the relative quality of the results provided by the heuristics without an explicit knowledge of the optimal solution, an analysis of the frequency with which each heuristic yielded the best and worst result was made. In this analysis, given a group of nets with equal number of terminals, the proportion of best and worst results given by each heuristic was computed. The results of this analysis are summarized in Figures 4.4 and 4.5 for the best and worst results, respectively.

From Figure 4.4 it is observed that for nets with 3 and 4 terminals both the DMH and the MBH provided the best result in more than 85% of the cases. For nets with five or more terminals the MBH consistently provided the best result most of the time. This behavior in the frequency of best solutions is very significant as it indicates that the skew in the estimation error of the MBH will be smaller than that of the rest of the evaluated heuristics. It must also be noted that although in typical circuits most nets have 4 or less terminals, it is also true that nets with 5 or more terminals are more likely to have larger capacitances than those with 4 terminals or less. Moreover, the analysis presented in Section 4.1 indicates that nets with 5 or more terminals are not less active than those with a lesser number of terminals. Therefore, the estimation of their wire requirements should not be less accurate, because of their likelihood of becoming critical power dissipating nets

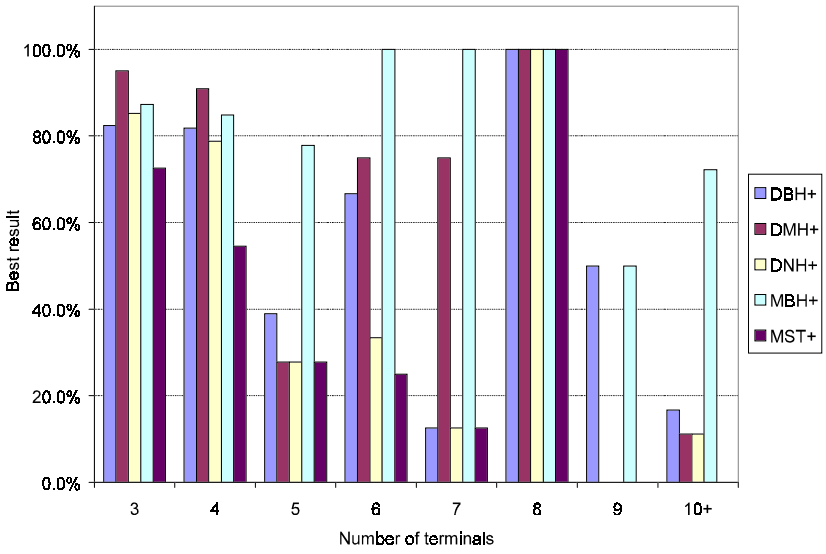


Figure 4.4. Proportion of best solutions given by each heuristic.

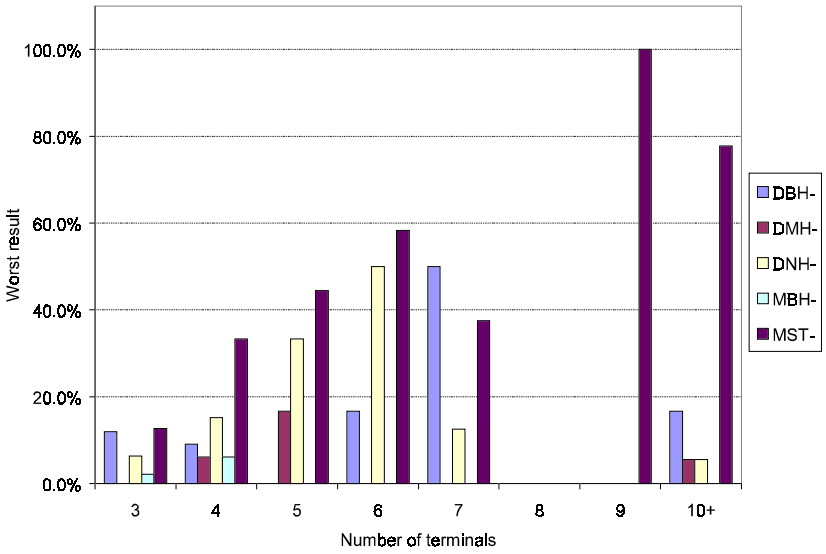


Figure 4.5. Proportion of worst solution given by each heuristic.

during placement. This reasoning then points to the MBH as the most appropriate wire length estimator for the low-power placement application.

The remaining heuristics, in general performed from average to poor with the MST being the worst. The DBH, DNH, and MST were found to provide good solutions in nets with a low count of terminals. However, they have a growing pattern of providing the worst solution as the terminal count per net increases. Figure 4.5 shows this behavior with the proportion of times that each heuristic provided the worst estimate as the number of terminals per net increased.

To assess the relative speed of the heuristics, their average running times were recorded for each set of nets with equal number of terminal points. Table 4.2 summarizes these statistics, obtained from the execution of the heuristics in a Sun Sparc Ultra 1. The results show that the MST, as expected by its reduced time complexity, provided the fastest solutions.

It can also be observed that although all distance-based heuristics have the same order complexity, the DNH is faster than others in its class. A close order analysis of its time complexity shows that the polynomial describing its time requirements has smaller coefficients than those in the other three heuristics. The time requirements of the MBH, DBH, and DMH were in the same neighborhood of values.

Table 4.2. Running time of the heuristics in ms.

Terminals per net	Heuristic				
	DBH	DMH	DNH	MBH	MST
3	19.2	10.7	0.5	19.5	0.3
4	4.8	3.5	0.5	5.2	0.2
5	170.7	115.5	3.9	272.9	1.7
6	52.5	29.8	2.2	78.7	0.7
7	313.2	131.8	4.8	227.5	1.4
8	1.7	1.8	0.8	1.5	0.1
9	80.6	76.3	9.2	117.3	1.8
10+	2397.4	1492.2	183.7	2149.2	7.0

4.6 Summary

Techniques for efficiently estimating interconnection networks in a power optimizing physical design environment have been studied. Given the accuracy constraints of the application framework, techniques falling in the class of greedy Steiner minimum tree (SMT) estimators were selected as the study target. Two new heuristics were proposed and evaluated: the median biased heuristic (MBH) and the diameter biased heuristic (DBH).

Algorithms for transforming macro block netlists into appropriate graphs were implemented. To limit the search space of the estimation heuristics, selected graph reduction techniques were adapted and implemented. These reductions were found to be both time efficient and powerful. Results indicate average deletions of over 95% of Steiner candidates in more than 90% of the cases.

The results from the SMT approximation heuristics favored the MBH in the consistency of quality results, a desired characteristic in the application framework of the estimators. Thus the MBH offers estimates with the minimum skew over the other heuristics for all tested network sizes.

CHAPTER 5

Implementation of the Power Optimizing Placement Tool

This chapter describes the implementation of the SAPPO tool. SAPPO stands for Simulated Annealing-based Placer for Power Optimization. The chapter begins with an introduction to the input format accepted by the program, followed by a description of the internal layout representation used in the program. Section 5.3 describes the tasks completed in the implementation of the simulated annealing heuristic. These include the design of the cost function and the annealing schedule. Special emphasis is placed on the description and analysis of a new algorithm developed for the handling of rectilinear shapes. Another area of emphasis is a description of the process followed in the calibration of cost function parameters and the annealing schedule. Results from calibration experiments are presented to assess the consistency and stability of the schedule.

5.1 Input Format

The SAPPO program accepts as input the specification of a circuit layout which includes the geometry and connectivity information of its modules, as well as the switching activity of its networks. Given that no standard netlist format was found that could support power optimization goals, a new physical layout netlist format was developed. This new format was based on the well known YAL netlist language used in MCNC benchmarks. It

extends the capabilities of its predecessor to support power optimizing physical design. The switching activity information embedded in the LPY netlist format can be obtained from the functional description of the circuit through the use of the previously selected switching activity estimator.

In addition to the netlist description, the input format must include a second set of data specifying design rules such as the electrical characteristics of the routing layers used in the layout, their minimum wire widths, inter-wire spaces, and constraints for the overall layout area among others. Appendix A describes in detail the extensions made to the base language to create the low-power YAL (LPY) format, as well as the rules and constraint specifications accepted by the SAPPO tool.

5.2 Internal Layout Representation

The input layout specification accepted by the placement tool is internally split into two data sets which capture the geometry, the inter-module relationships, and the electrical characteristics of the layout. The first data set, called the *line command set*, is a line-based representation of the layout which uses several line types to identify the boundaries of all modules and all interconnecting points. The second data set, called the *network list*, contains the connectivity among modules and the power parameters of each net. The network-module associations are made through a set of line commands representing the terminal sets joined together by a network.

The line-based representation was selected for several reasons, which include:

- *The speed of the line segment generation process.* The process can be completed in time $O(m)$, where m is the sum of the number of module sides plus terminal points in the layout.

- *The simplicity in performing geometric manipulations.* Module transformations involving displacement, orientation changes, and aspect ratio changes can be efficiently carried in a time proportional to the number of sides and terminals of the modules

involved in the operation.

- *The suitability of the line command set to implement line sweep-based overlap area computation algorithms.* The specific implementation in this research represents an improvement over previous approaches to this problem by handling general rectilinear shapes as single objects, without having to break them down into constituent rectangles. This overlap extraction algorithm was also used in the development of a channel pre-allocation mechanism in which the placement process produces sufficient inter-module space to route all terminals associated to neighboring blocks.

- *The support it provides to other algorithms.* The line-based representation allowed for the effective use of a line sweep mechanism as the basis of the algorithm that generates input graphs to the Steiner approximation heuristics.

A line command set is an instance-indexed list of line units representing module boundaries and terminals. Indexing of lines is made through an AVL tree [111]. Six basic command *types* are used. The first four: left, right, top, and bottom, denoted LC, RC, TC, and BC respectively, represent module sides. The last two commands, vertical and horizontal, denoted VC and HC respectively, represent terminal related lines. The designation of VC or HC to a given command is made depending on the unobstructed direction of a line escaping from the associated terminal.

Each line in a command is specified through a triplet (i, j, k). In horizontal lines, i and j represent the starting and ending abscissas of the line while k represents its ordinate. For vertical lines, i represents the abscissa while j and k represent the bottom and top ordinates.

A network list is an instance name indexed list of terminal commands associated to a common signal name and sharing a single switching activity value. Networks contain localized terminal commands which have received their (i, j, k) triplet after the modules have been assigned a specific location and orientation. Additionally, in the graph-based wirelength estimators, each entry in the network list contains the set of escape lines associated to the group of terminals joined by the net.

The main advantage of using a line-based representation of modules resides in the reduced time complexity of the algorithms used in the construction of the layout graphs required by graph-based wirelength estimation mechanisms and in the computation of module overlaps in intermediate placement solutions. The process of constructing and manipulating a layout graph was discussed in Chapter 4. Section 5.3.1 below describes the algorithms developed for computing module overlaps and performing the routing channel pre-allocation.

5.3 Annealing Algorithm

Simulated annealing, as previously established, was selected to be the optimization mechanism used by the SAPPO tool. The discussion presented in Chapter 2 identified three central tasks in the implementation of a simulated annealing-based placement strategy. These were the design of the cost function, the perturbation function, and the annealing schedule. This section discusses the implementation of these tasks in the SAPPO tool.

5.3.1 Cost Function

The cost function of the SAPPO tool was designed to reflect in a balanced way the different factors that measure the quality of a placement solution. In the current implementation, these factors are, in order of importance, the switched length, the amount of module overlap, the number of blocked terminals, and a penalty for insufficient inter-module space. The switched length, according to the discussion presented in Chapter 3, reflects the power cost of the current solution. The remaining factors represent constraints introduced into the objective function to guarantee the routability of all nets in the final solution. Other constraints, such as the maximum length of one or several delay critical nets, the length of the maximum signal path in the circuit, or limitations for power critical nets can be easily accommodated in the current implementation through the inclusion of appropriate penalties to the cost function.

The general expression for the total cost of a placement solution S, denoted $C(S)$, is

obtained from the sum of two components as

$$C(S) = C_1(S) + C_2(S). \tag{5.1}$$

Components $C_1(S)$ and $C_2(S)$ represent the costs related to the switched length and the overlap of the current solution S. The value of $C_1(S)$, obtained through equation (5.2), evaluates the switched length and the penalty assigned to nets detected as unroutable in the current solution.

$$C_1(S) = \sum_{j \in N_1} \alpha_j L_j + \sum_{k \in N_2} \alpha_k B(n_k). \tag{5.2}$$

The first term in the sum of equation (5.2) represents the switched length of the set of routable nets N_1 in S. Factor $\alpha_{(.)}$ is the switching activity of the net, and L_j its estimated wirelength. The second term assigns a penalty length to each unroutable net in the current solution. Denoting the set of unroutable nets in S by N_2, the penalty of a net $n_k \in N_2$, represented $B(n_k)$, is obtained as

$$B(n_k) = \zeta \cdot SPM(S) \cdot |T_{n_k}|, \tag{5.3}$$

where ζ is a scaling constant, $SPM(S)$ represents the semi-perimeter of the bounding rectangle of the current solution, and $|T_{n_k}|$ is the terminal count of the net being evaluated. The purpose of the scaling constant ζ is to adjust the weight of the penalty to not overload the cost function. The semi-perimeter of the core area was included to ensure that the penalty will automatically scale with the dimensions of the layout, while the terminal count ensures a linear growth of the penalty with the net size.

The second component of the global cost, $C_2(S)$, introduces the penalties associated with the overlap of modules and inter-module spacing. The expression used in the evaluation of this cost component is

$$C_2(S) = \psi \cdot \sqrt{(TO(S) + AO(S)) \cdot C_1(S)}. \tag{5.4}$$

In the evaluation of $C_2(S)$, ψ represents a scaling constant, $TO(S)$ is a function returning the overlap penalty among modules, and $AO(S)$ returns the penalty associated with violations to the minimum inter-module space. As in the case of ζ, the purpose of ψ is to adjust the weight of the penalty in such a way that it does not overload the cost function. Function $TO(S)$ measures the total overlap resulting from the assignment S of a set of locations L to the set of modules M in the circuit being placed. Given a core boundary constraint and an assignment S, the measure returned by $TO(S)$ includes the overlap area among modules as well as the overlap of modules with the exterior of the core boundaries.

Function $AO(S)$ measures the penalty associated with violations to the minimum inter-module space in the form of an overlap area. In doing so, the boundaries of each internal module are outward augmented by an amount of space equivalent to the number of terminals in each side of the module times the sum of the wire width and the inter-wire space of the routing layer. The boundaries of the layout are modeled as a square hole in a module which extends from minus to plus infinity in each direction. With this model, the augmentation of the core boundaries occurs inward, opposite to that of general modules. Figure 5.1 shows a sample layout with the augmented dimensions of its internal modules and core boundaries.

This inter-module space model assumes that each wire connecting to a module has exclusive use of a routing track spanning the entire side of the module. Also, one additional routing track is allocated on the right and bottom sides of each module, effectively creating a minimum inter-module space for general routing of the layout. Although this model represents the worst case of a pair-wise inter-module space allocation mechanism, for simplicity reasons, it does not consider routing congestion issues which arise around the center of the layout. However, it is expected that when the placed layout is taken to the routing stage, the over-allocation produced by the augmentation mechanism will compensate the for extra requirements of congested areas.

Once the module boundaries are augmented as indicated above, violations to inter-module spaces can be measured as overlap of their augmented areas. This condition is illustrated in Figure 5.1 by the dark area in the augmented area between modules m_1 and

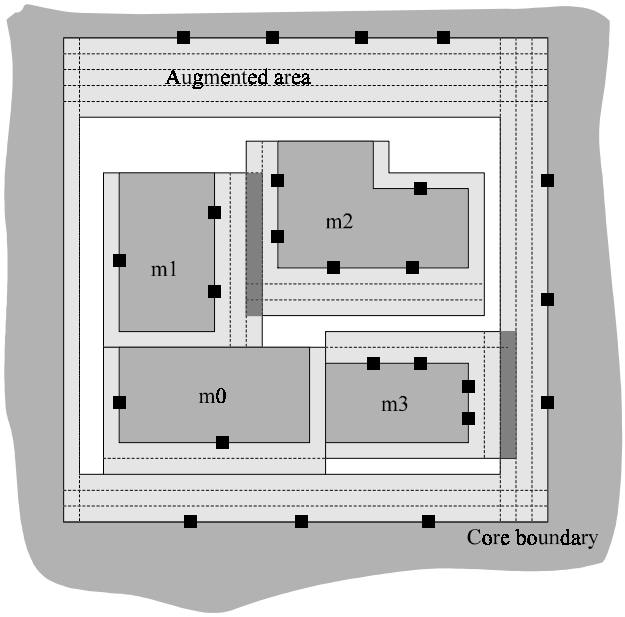


Figure 5.1. Sample layout with augmented dimensions.

m_2, and m_3 and the core. It should be noted that since $AO\,(S)$ is a measure of augmented overlap, it will also contain the quantity measured by $TO\,(S)$. However, experimentation has indicated that the redundant inclusion of $TO(S)$ made it possible for the cost function to differentiate between overlaps among only the augmented portions of a set of modules versus overlaps among the true module areas.

The similarity in the evaluation of functions $TO\,(S)$ and $AO\,(S)$ allowed for their implementation to be based on a common line search algorithm. Details of the implementation of these functions are discussed in Section 5.3.1.

In addition to the balancing constant and overlap functions, the value of $C_2\,(S)$ is dependent on the switched length costs represented by $C_1\,(S)$. This dependence was introduced to increase the relative weight of the overlap penalties with respect to the switched length costs when both switched length and overlap costs are small. The square root was introduced to drop the quadratic complexity of area measurements over the linear scale of the layout dimensions. This was necessary to ensure that in layouts with a fine grid resolution,

where linear measures grow large, the quadratic weight of the overlap does not overcome the switched length costs which grow linearly with the layout dimensions.

The relative weights of all factors making up the cost function have to be balanced in such a way that at the end of the annealing process the switched length component has been minimized, while the cost of all other penalties is reduced to zero. This balance should, however, allow for the coexistence of penalties during most parts of the annealing process, making them converge to zero only close to the end of the simulation. Experiments have shown that an early disappearance of penalties, especially those due to module overlaps, tends to produce poorer results than those obtained when penalties are allowed to coexist with the main optimization objective for most of the annealing process. Figures 5.2 and 5.3 illustrate typical cost behaviors for these situations.

The cost function in Figure 5.2 was set up to over-penalize the overlap. As a consequence, at an early point during the annealing process, intermediate solutions containing overlaps became unacceptably expensive. This forced the algorithm to enter a phase where only overlap-free configurations were accepted. Then, once the overlap disappeared, the reduction rate of the central objective was considerably slowed down. In Figure 5.3, the relative weights of the cost components had a balance which allowed the penalties to coexist with the central objective for a longer period of time. In this case, the initial slope of the reduction rate was maintained longer and the overall final solution went from 84170 to 69098, an improvement of 18% over the previous case.

The current implementation of the cost function has experimental determined values of $\zeta = \frac{1}{3}$ and $\psi = \frac{1}{3\sqrt{2}}$. The value currently assigned to ζ showed that at high temperatures, penalty $B(\cdot)$ was not too large to reject all moves containing blocked terminals. But as the global cost and temperature decrease, the relative cost of blocked nets is larger, making it more unlikely to accept solutions with unroutable terminals. The selected value of ψ corresponds to cases where the overlap penalty was held the longest, without reaching the final solution.

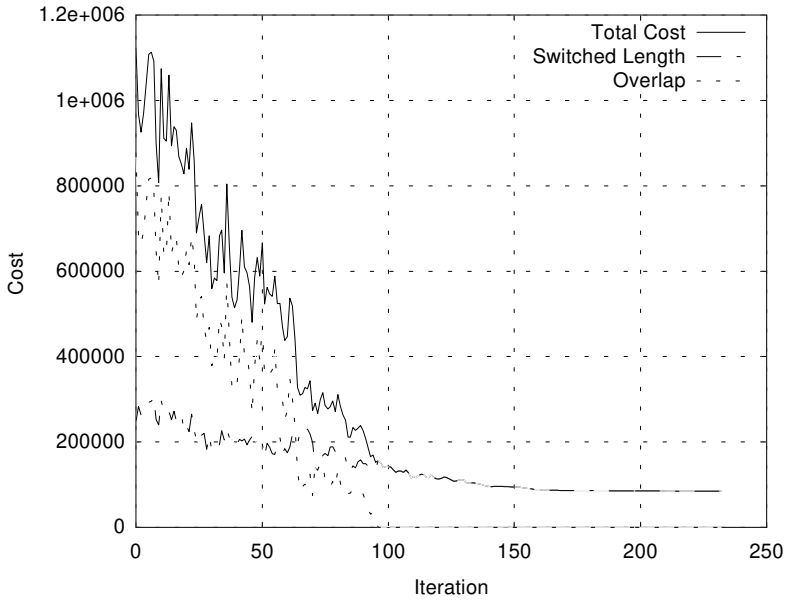


Figure 5.2. Cost behavior with early penalty removal.

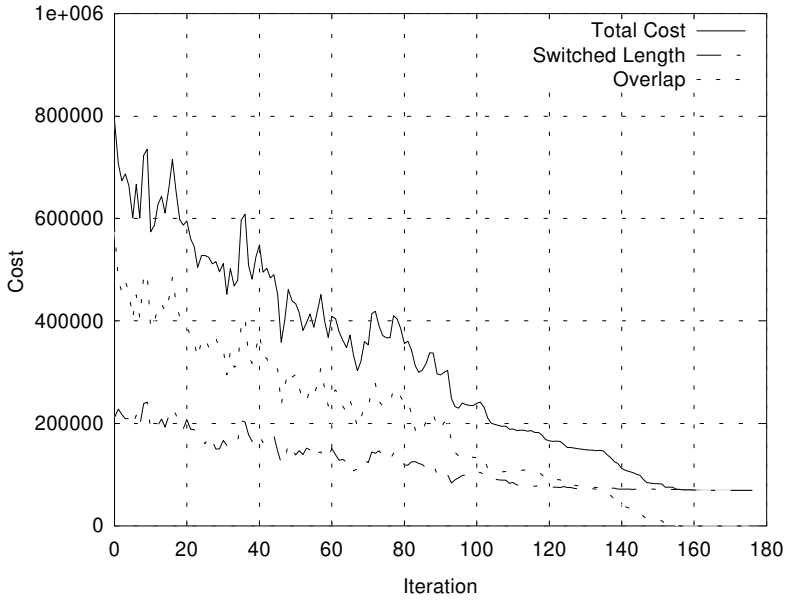


Figure 5.3. Cost behavior with late penalty removal.

Overlap Computation

In order for a placement solution to be feasible, it should not contain module overlaps. However, it has been reported that in iterative improvement mechanisms, allowing overlaps in intermediate solutions leads to better final placements [99], [112]. Overlap computation accepting such an intermediate solution was found to be performed in different ways, depending on the types of shapes supported in the layout. A few representative approaches are discussed below.

Onodera, *et al.* use an analytical overlap computation method based on a set of equations describing the topological relationships between pairs of modules [113]. This method requires $O\left(p^2\right)$ computations for a layout consisting of p modules. Their computation model was designed to only handle rectangular blocks.

Sechen, in the TimberWolf program, uses a bin-based approach which supports general rectilinear shapes [112]. Support for general rectilinear blocks is provided through a mechanism that breaks non-rectangular modules into an equivalent covering set of rectangular tiles. The overlap model divides the layout area into a grid of rectangular bins indexed by their row-column position. When a module is assigned to a specific location, a record is created on the bins intercepted by the rectangular tiles of the module. If tiles from other modules intercept any of the bins already intercepted by another block, an overlap computation is performed with the cells occupying the shared bins. This model requires in the worst case a time $O\left(p^{\frac{5}{3}}\right)$ if a module intercepts all bins. In a later revision of the model, Swartz found that it might fail to detect overlaps for small modules [99]. A modification was introduced which refined the computation of the bin size to reduce the possibility of an occurrence of this error.

Several other macro block placement techniques were also analyzed, but none of them considered either non-rectangular shapes or intermediate overlapped solutions [57], [114–116].

The SAPPO tool, as it was inferred earlier in the discussion of the cost function components, was designed to accept intermediate configurations with module overlaps through

the assignment of a properly balanced penalty cost. The overlap computation mechanism selected was however, different from any of those discussed above. A method based on line search contour detection algorithm was implemented instead.

Efficient line search-based algorithms have been reported in the literature for several planar applications in computational geometry. These include counting and reporting intersecting rectangles, the *batched range search* problem, and finding the contour of a set of intersecting rectangles, among others [117-119]. The direct application of these algorithms on rectilinear objects was found to be limited to rectangular shapes. Non-rectangular shapes could only be handled by first decomposing them into an equivalent set of non-overlapping tiles. This decomposition approach, although practical, increases the number of objects to be handled by algorithms that manipulate general rectilinear shapes. Moreover, in placement applications, it also requires the introduction of additional constraints into the objective function to keep the constituent rectangles from being separated by perturbation mechanisms.

The overlap computation mechanism implemented in the SAPPO program is based on a new contour detection algorithm for rectilinear objects which does not require breaking-up non-rectangular shapes into constituent tiles. This new algorithm starts on the work reported by Lipski and Preparata for rectangular shapes [118], and expands by introducing algorithmic modifications which allow for the direct handling of general rectilinear shapes without increasing either the average time or space complexity of the former algorithms. The descriptions of the contour and overlap computation algorithms follow.

Let $M = \{m_1, m_2, \cdots, m_p\}$ be a set of general rectilinear modules which has received a placement S in a bounded plane, receiving for each $m_i \in M$ a unique location l_j. Then, the union of the domains of the elements in M, denoted $F = D_1 \cup D_2 \cup \cdots \cup D_p$ consist of one or more disjoint connected pseudo-domains, whose contour $\mathbb{C} = \{C_1, C_2, \cdots, C_b\}$ is a set of disjoint cycles, each of which is representable by a set of alternating vertical and horizontal module commands. Each cycle C_i is assigned a direction such that it will rotate clockwise on the boundaries of a hole, and counter-clockwise otherwise. The contour problem is that

of finding set $\mathbb{C}$.

The algorithm used in the computation of $\mathbb{C}$ consists of two separate phases in which the vertical and horizontal edges of the contour cycles are respectively obtained. The first phase is the critical stage in the handling of general rectilinear shapes. The second phase follows the same approach reported in [118] for extracting the set of horizontal lines. A description and analysis of each phase follows.

Phase I: Detection of Verticals. In the first phase, the detection of the vertical segments of the contour cycles is made through a scan of the vertical lines of each shape. This is achieved by symbolically moving a vertical line from left to right over the layout space. As portions of the scan line are occluded or uncovered by modules, the vertical edges of the contour cycles are detected and reported. Algorithm 5.1 shows the sequence of operations performed in the first phase of the contour detection algorithm.

Algorithm 5.1 (Contour, Phase I)

Input: *H and V = Horizontal and vertical commnad sets of M.* $|V| = |H| = \frac{b}{2}$

Output: *Vertical line command set of* $\mathbb{C}$

1. $W \leftarrow H$ sorted in ascending order of k (ordinates) without repetitions;
2. $ST \leftarrow$ segment tree of M made using W;
3. $V' \leftarrow V$ sorted in ascending order of i, *type*, j, and k;
4. $\forall v \in V'$ do:
   - 4.1. If $(v.type = LC)$ then $Stack \leftarrow$ **Insert**$(v.bottom, v.top, ST, af \leftarrow$ FALSE$)$;
   - 4.2. Else $Stack \leftarrow$ **Delete**$(v.bottom, v.top, ST, af \leftarrow$ FALSE$)$;
5. Return *Stack*;

Performing the scan requires sorting the set of vertical edges of all modules in increasing order of their abscissas. For a layout with b module sides, this step can be completed in $O(b \cdot \log b)$ time and with space complexity $O(b)$. The detection of which portions of the module sides contribute to the set of vertical edges of the contour cycles is made with the aid of a *segment tree*.

A segment tree is a data structure based on a perfectly balanced binary tree, proposed by Bentley and Woods, to represent a line segment made up of a number of smaller sub-segments whose end points are known in advance [117]. Each vertex of a segment tree contains four fields: *bottom*, *top*, *cover*, and *status*, in addition to the pointers to its *left* and *right* children. Fields *bottom* and *top* represent the end points of the interval represented by the node, *cover* is a counter of the number of segments that cover the node, and *status* is a flag indicating whether the coverage of the current node is *full*, *partial*, or *empty*. A full status implies that at least one segment covers the current vertex from bottom to top. A partial status indicates that at least one child of the current vertex is covered, while empty indicates that neither the current vertex or any of its children is covered.

The construction of a segment tree, invoked in Step 2, is completed as follows. Given an ordered set of intervals W making up a line segment, a new vertex (*root*) is created for the entire set W. If the number of elements in W is larger than one ($|W| > 1$), the input set is partitioned into two halves W_1 and W_2 such that $|W_2| - |W_1| \leq 1$. Subsets W_1 and W_2 are assigned to the left and right child of root, respectively, and the same procedure is then repeated on each, until subsets with only one interval are created. As can be seen, the whole process can be completed in time $O(|W|)$ with the same space complexity. Given that $|W| = \frac{b}{2}$, the time and space complexity of Step 2 is $O(b)$.

In the traversal of the layout, finding line commands corresponding to a left or right edge might increase or decrease the coverage of one or several segments. These events, symbolized by procedures **Insert** and **Delete** respectively, will change the status and coverage of nodes in the segment tree. If the coverage of a certain node increases from 0 to 1, or decreases from 1 to 0, and none of its ancestors is fully covered, the interval represented by the node will contribute to a contour cycle.

The parameter af used in the invocation of **Insert** and **Delete** keeps these procedures from reporting false contour contributions deep inside the segment tree. Parameter af indicates whether or not an ancestor of the current root in the tree was detected to have full status. The implementation of the **Insert** procedure is outlined in Algorithm 5.2.

Algorithm 5.2 (Insert)

Input: (b, e) = *End points of line command, root = Current vertex,* af = *Ancestry status*

Output: *Contribution of line command to the contour*

1. If ($root.status$ = FULL) then $af \leftarrow$ TRUE;
2. If ($b \leq root.bottom$ and $e \geq root.top$)
   - 2.1. If (af = FALSE) then $Stack \twoheadleftarrow$ **Report**($root$);
   - 2.2. $root.cover \leftarrow root.cover + 1$;
3. Else
   - 3.1. If ($b < root.left.top$) then $Stack \twoheadleftarrow$ **Insert**$(b, e, root.left, af)$;
   - 3.2. If ($e > root.right.bottom$) then $Stack \twoheadleftarrow$ **Insert**$(b, e, root.right, af)$;
4. **Update**($root$) ;
5. Return($Stack$);

In the above implementation, *Stack* represents an external stack used for reporting contributions. In the operation of Algorithm 5.1, all contour commands pushed ($\twoheadleftarrow$) onto *Stack* are retrieved whenever there is a change in the input command type or in the abscissa value of the scan process.

The general implementation of the **Insert** procedure is very similar to that for rectangular shapes, except in two details. First is the conditioning of the ancestry of the current root in Step 2.1. Second is the form in which children nodes are tested for recursive insertion in Steps 3.1 and 3.2. This indexing avoids the need for a contents associative search required in the former implementation when the normalized ordinates are used to index the segments.

The reporting procedure invoked in Step 2.1 of the **Insert** routine places on the stack a set of line segments corresponding to the contribution of the current node to the contour. Algorithm 5.3 outlines the implementation of **Report**.

Algorithm 5.3 (Report)

Input: *root = Current vertex*

Output: *Contribution of current vertex to the contour*

1. If ($root.status$ = EMPTY)
   1.1. $line \leftarrow (root.bottom, root.top)$;
   1.2. $line \leftarrow$ **Join**$(line, Stack)$;
   1.3. $Stack \twoheadleftarrow line$;
2. Else if ($root.status$ = PARTIAL)
   2.1. **Report**$(root.left)$;
   2.2. **Report**$(root.right)$;
3. Return$(Stack)$;

When reporting the contribution of general rectilinear shapes, a single segment can be reported as several intervals. The objective of the **Join** function called in Step 1.2 is to merge segments already inserted into the stack which might be colinear with *line*. The joining operation retrieves from *Stack* any segment identified to be lower or upper colinear to *line*, joins them together, and returns to *Stack* the equivalent segment. This procedure fixes a problem identified in the former version of the contour algorithm which fails to properly report the contribution of a set of stacked shapes aligned on either their leftmost or rightmost sides.

The new implementation of **Join** has a worst case time and space complexity $O(b)$ if all shapes are stacked and aligned on one of their vertical sides. Considering that **Join** will not merge more than $\frac{b}{2} - 1$ intervals in one entire recursive invocation of **Insert**, and that the rest of the operations in the insertion process are completed, in the worst case, in time $O(\log b)$, the resulting time and space complexity of the entire insertion process is $O(b + \log b)$, which is equivalent to $O(b)$.

The **Update** function, called in Step 4 of the **Insert** procedure, is outlined in Algorithm 5.4. Its objective is to update the status of vertices affected by a segment insertion, based on the resulting coverage count of each vertex and its children.

Algorithm 5.4 (Update)

Input: *root = Current vertex*

Output: *None. The input vertex* status *field is updated.*

1. If ($root.cover$) then $root.status \leftarrow$ FULL;
2. Else
   2.1. If ($root.left$)
      2.1.1. If ($root.left.status \neq$ EMPTY or $root.right.status \neq$ EMPTY)
         2.1.1.1. $root.status \leftarrow$ PARTIAL;
      2.1.2. Else $root.status \leftarrow$ EMPTY;
   2.2. Else $root.status \leftarrow$ EMPTY;
3. Return;

The extension of the contour extraction algorithm to support general rectilinear shapes required a completely redesigned **Delete** procedure. When only rectangular shapes are supported, every call to **Delete** will uncover from the segment tree an interval of length equal to some other interval previously covered by **Insert**. For non-rectangular shapes, the segments removed by **Delete** are not necessarily of the same length as any other previously covered with **Insert**. Consider the cases illustrated in Figure 5.4.

A left-to-right scan of module m_1 will insert segments a and b into the tree, but will delete segment c, which is neither the length of a or b. The converse situation will happen

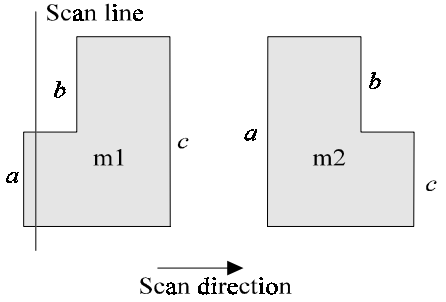


Figure 5.4. Scan of non-rectagular modules.

in a similar scan of module m_2. When the former version of the algorithm is presented with either of these cases, the segment tree is corrupted. To properly handle these cases, the new **Delete** procedure introduces the operations split and uncover. These new functions break down or uncover, as required, segments in the tree to correctly report the contribution of each module side to the contour. After either of these operations is completed, the status in the segment tree will be updated to reflect the changes. Algorithm 5.5 outlines the implementation of the new **Delete** procedure.

Algorithm 5.5 (Delete)

Input: (b, e) = *End points of line command, root = Current vertex,* af = *Ancestry status*

Output: *Contribution of line command to the contour*

1. If ($b \leq root.bottom$ and $e \geq root.top$)
   - 1.1. $root.cover \leftarrow root.cover - 1$;
   - 1.2. If ($root.cover < 0$)
     - 1.2.1. $root.cover \leftarrow root.cover + 1$;
     - 1.2.2. **Uncover**$(b, e, root)$;
   - 1.3. Update$(root)$;
   - 1.4. If ($af =$ FALSE) then $Stack \Leftarrow$ **Report**$(root)$;
2. Else
   - 2.1. If ($root.status$) then **Split**$(root)$;
   - 2.2. If ($root.cover$) then $af \leftarrow$ TRUE;
   - 2.3. If ($b < root.left.top$) then $Stack \Leftarrow$ **Delete**$(b, e, root.left, af)$;
   - 2.4. If ($e > root.right.bottom$) then $Stack \Leftarrow$ **Delete**$(b, e, root.right, af)$;
   - 2.5. Update$(root)$;
3. Return$(Stack)$;

The implemented **Delete** procedure will correctly uncover and report rectangular and non-rectangular shapes. The case when a previously inserted large left side is being uncovered by a set of smaller right sides, as in the case of m_2 above, is properly handled in Step 2.1 with the **Split** function. This function partitions a large covered *root* by reducing its

cover by one, increasing by the same amount the covers of its two immediate children, and updating accordingly the status of the three vertices involved. This operation keeps the tree from being corrupted with negative cover values.

The case posed by m_1 in Figure 5.4 could still cause a node cover to become negative. In this case the **Uncover** procedure, invoked in Step 1.2.2, searches the segment tree rooted at *root* to uncover a set of previously inserted sub-segments covering the interval spanned by the large right segment. Algorithm 5.6 outlines the implementation of the **Uncover** function.

Algorithm 5.6 (Uncover)

Input: (b, e) = *End points of line command, root = Current vertex*

Output: *None. Uncovers a segment set spanning* $[b, e]$

1. If ($b \leq root.bottom$ and $e \geq root.top$ and $root.status =$ FULL)
   1.1. $root.cover \leftarrow root.cover - 1$;
2. Else
   2.1. If ($b < root.left.top$) **Uncover**$(b, e, root.left)$;
   2.2. If ($e > root.right.bottom$) **Uncover**$(b, e, root.right)$;
3. **Update**$(root)$;
4. Return;

The worst case time complexity of the **Uncover** procedure occurs when it is invoked from the root of the segment tree and the set of covering sub-segments are in the leaves of the tree. In this case the algorithm would take $O(b \cdot \log(b))$ time to complete.

When analyzing how these requirements would manifest in the calling procedure, it can be observed that although **Delete** has a recursive execution, an invocation to **Uncover** with the worst case time complexity can occur at most once in the entire execution of **Delete**. In cases where multiple invocations to **Uncover** could occur, the worst case would be when uncovering a segment whose length is covered by $\frac{b}{2} - 1$ leaves of the tree (recall that there will be at most $\frac{b}{2}$ sub-segments in the tree). In this case, the **Uncover** algorithm would be called $\log\left(\frac{b}{2}\right) - 1$ times, with a decreasing sub-tree depth at each new call. The

resulting complexity of calls to **Uncover** would be in this case given by

$$\sum_{i=1}^{\log\left(\frac{b}{2}\right)-1} \left[\frac{b}{2^{i+1}} \cdot \log\left(\frac{b}{2^{i+1}}\right)\right] < b \cdot \log(b) . \tag{5.5}$$

This complexity, as indicated in equation (5.5), is bounded by the worst case single call complexity obtained earlier.

Considering the total time requirements of all functions called during the execution of the new **Delete** procedure, the resulting time complexity would be

$$O(2b \cdot \log(b) + \log(b)) = O(b \cdot \log(b)) . \tag{5.6}$$

It can be inferred from the above analyses that the worst case time complexity of the first phase of contour detection algorithm is dominated by the requirements of the scanning loop (Step 4 in Algorithm 5.1), which would run in $O\left(b^2 \cdot \log(b)\right)$ time.

The worst case complexity, however, is very unlikely to occur in practical circuits since it would require that all modules are non-rectangular and spanning one entire dimension of the layout. In reality, the dimensions of internal blocks are always smaller than those of the core, and non-rectangular modules are typically found in mixed layouts containing also mostly rectangular shapes. For these reasons, the average complexity would be more representative of the time requirements of this algorithm. In the average case the **Delete** procedure would execute in time $O(\log(b))$, since no call to **Uncover** will take time $O(b \cdot \log(b))$. As a consequence, the average complexity of the first phase of the contour algorithm would result $O(b \cdot \log(b))$.

Something to also be considered is that the additional operations performed under the new algorithm on non-rectangular shapes are less expensive than decomposing them into multiple rectangles and then recombining. This is especially true in iterative placement applications, where the decomposition is carried to the solution perturbation mechanisms, where they will exponentially increase the size of the solution space to be searched.

Phase II: Extraction of Horizontals. The process of extracting the horizontal sides produces the actual contour cycles used in the overlap area computation. Given the set of verticals detected in the first phase, the horizontal extraction process proceeds as outlined in Algorithm 5.7.

Algorithm 5.7 (Contour, phase II)

Input: *Q' = Set of vertical contour commands.* $|Q'| = d \leq \frac{b}{2}$

Output: *Set of contour cycles*

1. $Q \leftarrow Q'$;
2. $\forall v' \in Q'$ with $v' = (i, j, k, type)$ do
   - 2.1. $v \leftarrow (i, k, j)$;
   - 2.2. $Q \leftarrow Q \cup v$;
3. $Q \leftarrow Q$ sorted in increasing order of k, i;
   - 3.1. (Resulting command sequence: $Q = \{v_1, v_2, \cdots, v_{2d}\}$)
4. $C \leftarrow \emptyset$; $\mathbb{C} \leftarrow \emptyset$ /* Single cycle and cycle list */
5. $\forall t, t = 1, 2, \cdots, d$ do
   - 5.1. Assume: $v_{2t-1} = (i, j, k, type)$ and $v_{2t} = (i', j', k', type')$;
   - 5.2. $h_t \leftarrow (j, i, i', \text{null})$; /* Type to be assigned */
   - 5.3. If $((v_{2t-1}.type = \text{LC}$ and $j < k)$ or $(v_{2t-1}.type = \text{RC}$ and $k < j))$
     - 5.3.1. $h_t.type \leftarrow$ BC;
   - 5.4. Else
     - 5.4.1. $h_t.type \leftarrow$ TC;
   - 5.5. $C \leftarrow C \cup (v_{2t-1}, h_t, v_{2t})$;
   - 5.6. If (h_t closes cycle C)
     - 5.6.1. $\mathbb{C} \twoheadleftarrow C$;
     - 5.6.2. $C \leftarrow \emptyset$;
6. Return($\mathbb{C}$);

The loop in Step 2 augments the command set by duplicating the vertical commands with their ordinates j and k exchanged. Sorting the resulting set by k and i produces an

ordering where the horizontal sides are delimited by the abscissa values of commands v_{2t-1} and v_{2t}, for $t = 1, 2, \cdots, 2d$, respectively. The type of horizontal command is given by the rule in Step 5.3, and the rotating direction is defined by the orientation of the leftmost of the lowest horizontals in the cycle. A cycle is closed when the newly created horizontal joins two vertical sides already in the cycle. The space complexity in this phase of the algorithm is determined by Step 2 to be $O(b)$, while the time complexity, determined by Step 3, remains $O(b \cdot \log(b))$.

The overall average time complexity of the complete contour extraction is $O(b \cdot \log(b))$ and its space complexity $O(b)$.

Area Computation. Once the contour cycles are known, the overlap area of solution S, denoted $\Gamma(S)$, can be obtained as

$$\Gamma(S) = \sum_{m_i \in M} |D_i| - \sum_{C_i \in \mathbb{C}} |C_i|, \tag{5.7}$$

where $|D_i|$ represents the domain area (of module m_i) and $|C_i|$ the area enclosed by cycle C_i. When computing the individual terms of each sum, the area of clockwise cycles are assigned negative values, while that of counterclockwise loops are assumed positive.

5.3.2 Perturbation Function

The block perturbation mechanism used in the SAPPO tool produces new placement solutions by introducing small incremental changes to a given configuration. The new solution is then evaluated with the cost function described in the previous section, and applied to the Metropolis criteria for acceptance.

The changes induced on a given solution by the implemented perturbation mechanism can be directed to affect one or two modules at a time, depending on the selected type of perturbation. Changes affecting a single module m_i, are:

- $MD1(m_i)$: Displacement of module m_i to a randomly selected new location preserving its current orientation.

- $MD2\,(m_i)$: Displacement of m_i to a randomly selected new location with an orientation change affecting its aspect ratio.

- $MD3\,(m_i)$: A randomly selected orientation change of m_i in its current location.

Perturbations affecting a pair of modules m_i and m_j at a time are:

- $MX1\,(m_i, m_j)$: Exchange the locations of modules m_i and m_j preserving their original orientations.

- $MX2\,(m_i, m_j)$: Exchange the locations of m_i and m_j, with an aspect ratio change applied to both modules.

The ratio of double to single module perturbations produced by the program is controlled by a pre-determined parameter. This parameter was experimentally determined by analyzing the variations it produced in the quality of the final solution of a test case with ten macro blocks. Several values between 0.5 and 0.04 were tested while leaving all other parameters unchanged. The best resulting ratio among those tested was 0.10. Table 5.1 summarizes the tests performed.

The implementation of the perturbation function begins by determining the number of modules to be perturbed and selecting them. If a single module is selected, an attempt is made with an $MD1$ type perturbation. If this operation is rejected, an $MD2$ move is attempted by changing the orientation of the selected module. If this move is also rejected, an $MD3$ perturbation is attempted. If any of the three attempts is successful, the perturbation is integrated into the current solution or otherwise left unchanged. When two modules are selected, the initial attempt is with an $MX1$ type move. If this fails, the

Table 5.1. Effects of the displacement to exchange ratio.

*Ratio*12	1/2	1/4	1/10	1/16	1/25
Final Cost	139817	116237	85413	107914	163309

aspect ratios of the selected blocks are changed to attempt an $MX2$ exchange. An outline of the implemented perturbation function is illustrated in Algorithm 5.8.

The function **RndInt** invoked throughout the procedure generates a random integer number uniformly distributed in the range specified by its two parameters. The range function invoked in Step 2.2 is a range limiter which keeps the process from generating large perturbations at low temperatures, when they are most likely to be rejected [94]. The limit imposed on the range of moves varies linearly with the logarithmic rate of the temperature changes. The expression used in the range limit computation (RL) is

$$RL_x = \epsilon \frac{\log_{10}(T_k)}{\log_{10}(T_0)} \cdot x_{\max}, \tag{5.8}$$

where T_0 and T_k represent the initial and current temperature values, respectively, $x_{\max}$ the width of the core area, and ϵ is an experimentally determined constant. The current implementation uses $\epsilon = 1.0$, a value found to allow a uniform exploration of the solution space when T_k is relatively close to T_0. At low temperatures, a lowest bound of $\min(9, 0.1x_{\max})$ layout units was assigned to RL_x. Equation (5.8) gives the formula used by **RangeX**. Function **RangeY** performs a similar computation based on the height of the core $y_{\max}$. Function **Accept** evaluates the cost of the changes suggested by the perturbation and returns and acceptance flag based on the Metropolis criteria.

Figure 5.5 shows a typical space exploration achieved by **Perturb**. The plots in Figure 5.5a and 5.5b show how the perturbation function uniformly explores the width and height of the layout at the beginning of iterations, when T_k is relatively high. As the process advances (lower temperature values), the range limiter reduces the scope of the space exploration to limit the search around the position established to be best for each module. The seven clusters identified in Figure 5.5c reveal the spatial exploration around the final locations assigned to the seven modules of the selected test case.

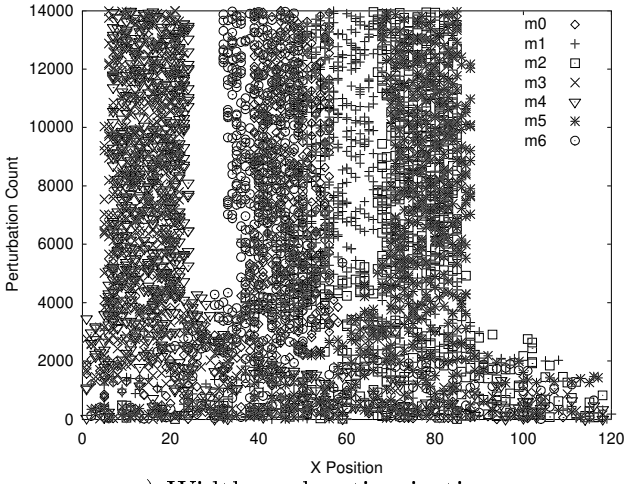
Algorithm 5.8 (Perturb)

Input: *Sol* $[NB]$ = *Input solution with* NB *input blocks*

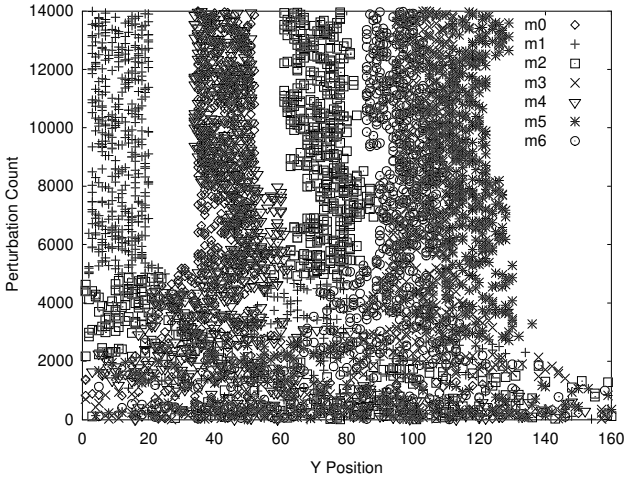
Output: *Perturbed or previous solution*

Variables: $NBTP$ = *Number of blocks to perturb,* bid = *Block ID,* $ornt$ = *orientation*

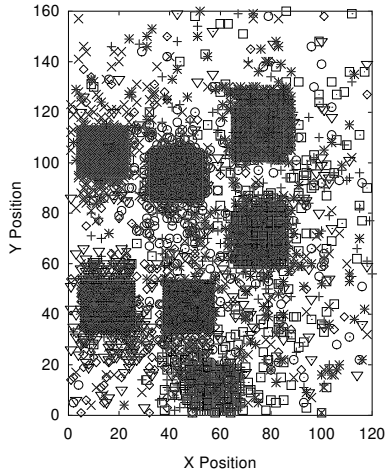
1. $NBTP \leftarrow \mathbf{RndInt}(1,2)$ with $prob(1) = 1 - Ratio21$;
2. If $(NBTP = 1)$
   - 2.1. $bid1 \leftarrow \mathbf{RndInt}(1, NB)$;
   - 2.2. $\{x, y\} \leftarrow \{\mathbf{RndInt}\,(\mathbf{RangeX}\,(Sol\,[bid1]))\,, \mathbf{RndInt}\,(\mathbf{RangeY}\,(Sol\,[bid1]))\}$;
   - 2.3. $ornt \leftarrow$ Current orientation of $Sol\,[bid1]$;
   - 2.4. If $(\mathbf{Accept}\,(Sol\,[\cdot]\,, bid1, (x, y, ornt)))$
     - 2.4.1. $Sol\,[bid1] \leftarrow (x, y, ornt)$; Return$(Sol\,[\cdot])$;
   - 2.5. $ornt \leftarrow \mathbf{RndInt}(1, 4)$;
   - 2.6. If $(\mathbf{Accept}\,(Sol\,[\cdot]\,, bid1, (x, y, ornt)))$
     - 2.6.1. $Sol\,[bid1] \leftarrow (x, y, ornt)$; Return$(Sol\,[\cdot])$;
   - 2.7. $ornt \leftarrow \mathbf{RndInt}(1, 7)$;
   - 2.8. $(x, y) \leftarrow$ Current coordinates of $Sol\,[bid1]$
   - 2.9. If $\mathbf{Accept}(Sol\,[\cdot]\,, bid1, (x, y, ornt))$
     - 2.9.1. $Sol\,[bid1] \leftarrow (x, y, ornt)$;
3. Else
   - 3.1. $\{bid1, bid2\} \leftarrow \{\mathbf{RndInt}\,(1, NB)\,, \mathbf{RndInt}\,(1, NB)\}$;
   - 3.2. If $(\mathbf{Accept}\,(Sol\,[\cdot]\,, bid1, (x_2, y_2, ornt_1)\,, bid2, (x_1, y_1, ornt_2)))$
     - 3.2.1. Make permanent the exchange into $Sol\,[\cdot]$;
     - 3.2.2. Return$(Sol\,[\cdot])$;
   - 3.3. $ornt \leftarrow \mathbf{RndInt}(1, 4)$;
   - 3.4. If $(\mathbf{Accept}\,(Sol\,[\cdot]\,, bid1, (x_2, y_2, ornt)\,, bid2, (x_1, y_1, ornt)))$
     - 3.4.1. Make permanent the exchange into $Sol\,[\cdot]$;
4. Return$(Sol\,[\cdot])$;



a) Width exploration in time



b) Height exploration in time



c) Layout area exploration

Figure 5.5. Space exploration performed by the perturbation function.

5.3.3 Annealing Schedule

The SAPPO tool was implemented around an adaptive annealing schedule where most parameters are computed using statistical properties from the problem to be solved. This approach dynamically fits the schedule to each individual problem. The design of such a schedule combines ideas and concepts discussed in several studies published on annealing schedules [90], [94], [95].

Melting Temperature

The initial temperature of the annealing process T_0 is obtained using a widely accepted criterion proposed by White establishing that the system is melted if $\sigma \ll T_0$, where σ is the standard deviation of the cost distribution [94]. To obtain a value of T_0 satisfying this criteria, an initial space exploration of the configuration is performed assuming that the temperature has an infinite value. This would lead to an acceptance of all proposed moves by the Metropolis criteria. It has been found that under these conditions the cost function of typical problems has a uniform distribution [95]. Thus, selecting a temperature high enough to accept with probability P a configuration whose cost is $k \cdot \sigma$ times worse than the current one, with k being an arbitrary constant, the initial temperature can be obtained as

$$T_0 = -\frac{k}{\ln(P)} \cdot \sigma. \tag{5.9}$$

The current implementation of the melting function in the SAPPO tool uses the values $k = 3$ and $P = 0.85$ in the evaluation of equation (5.9). The standard deviation of the cost distribution is computed after at least $2p$ moves have been made, where p is the number of blocks in the layout being placed.

Inner-loop Criterion

The number of iterations performed at each temperature value is dynamically determined using an equilibrium detection mechanism proposed by Huang *et al.* [95]. It establishes that once equilibrium is achieved, the ratio of the number of new states generated with

their costs within a certain range δ from the average cost μ to the total number of newly accepted states will reach a stable value χ. A normal distribution of the cost function is assumed. Thus, the ratio between the number of accepted states whose cost is in the range $(\mu - \delta, \mu + \delta)$, referred to as the *within count*, and the number of accepted states is given by $\text{erf}\left(\frac{\delta}{\sigma}\right)$, where $\text{erf}(\cdot)$ is the error function [120]. The value of δ should be a fraction of σ so that the cost of the final state in each iteration is close to that of μ.

Given the above ratio and a problem size measured by the number of blocks to be placed p, the parameters needed to specify the equilibrium condition would be a target *within count* β and a tolerance limit γ. The tolerance limit is the maximum number of states accepted outside the target range. Equilibrium is considered to be reached when the *within count* reaches β before the tolerance reaches γ. Otherwise both counts are reset to zero and the process is restarted. Considering that at low temperatures the target *within count* might never be reached, an upper limit B is set on the number of newly generated states. To guarantee the validity of statistics, the detection of equilibrium is not started until a minimum number G of states has been accepted. This additional condition requires an ultimate limit U for the generation of states at low temperatures when G might not be reached. The specific values assigned to the above set of parameters in the implementation of the SAPPO tool are controlled through a set of parameters specified at run time. A discussion on how these values are assigned is included below in Section 5.3.3.

Temperature Decrement Rate

The temperature decrement rate υ was experimentally determined through a process where the behavior of several test circuits was analyzed. Given the constraint that consecutive temperature changes had to be sufficiently small to allow for a quick re-establishment of quasi-equilibrium, an initial rate change of $\upsilon = 0.96$ was set and fixed for the entire process in all test cases. The experiment showed the following.

At high temperatures, quasi-equilibrium was easily obtained and the average solution cost decreased in almost all cases about 10% of its total reduction. It was deduced that this phase could tolerate a more aggressive temperature reduction rate without significantly

altering its overall behavior. This behavior was observed to persist for about 1.6 decades of temperature decrease, starting at the melting point.

At medium-high temperatures the solution cost exhibited its most drastic reduction and an increasing count in the number of attempted moves in the inner loop. This implied that during this period, larger temperature changes could make it harder to reach quasi-equilibrium. This phase was observed to last about two decades in most circuits. A region of medium-low temperatures was also identified, exhibiting a less drastic reduction in the solution cost and even longer inner loops. The cost behavior suggested that this zone could support a slightly faster temperature reduction scheme. This phase was observed to last for about one decade.

At low temperatures, virtually all temperature steps reached the ultimate limit U in the number of trials, each with very few accepted moves and a very small reduction in the average solution cost. The attainment of limit U in virtually every inner loop iteration indicates that equilibrium was being forced due to the extremely low acceptance ratio. This behavior is very costly in terms of time and lead to the conclusion that an aggressive temperature reduction scheme would positively impact the running time of the heuristic without a severe degradation in the quality of solution. These observations led to the implementation in the SAPPO tool of a temperature reduction strategy as illustrated in Table 5.2.

Table 5.2. Temperature reduction rule.

Temperature region	Reduction rate (ν)	Length in decades
High	0.87	1.6
Medium-high	0.96	2.0
Medium-low	0.85	1.0
Low	0.80	Remainder

Tests performed with the above strategy did not show a significant change in the overall

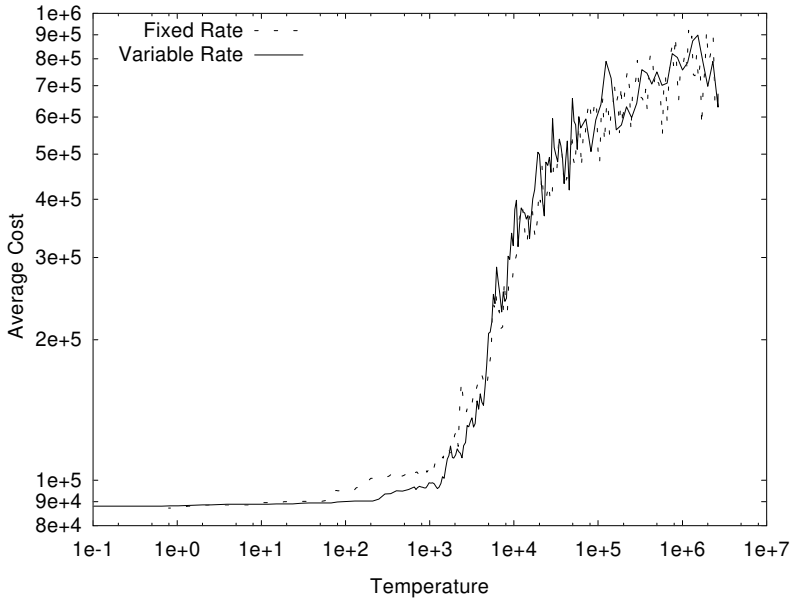


Figure 5.6. Solution cost vs. temperature for fixed and variable values of v.

behavior of the algorithm with respect to using the fixed reduction rate. Figure 5.6 shows the solution cost behavior before and after the application of the implemented strategy for a typical case.

Table 5.3 shows the average changes induced in the number of temperature iterations, attempted moves, final cost, and running time obtained with the new strategy on the test case. It can be observed that the new decrement strategy produced the same quality of solution while reducing the total annealing time by almost 50%.

Table 5.3. Effects of the temperature reduction rate.

	Iterations	Moves	Final Cost	Time (s)
Fixed	334	32471	87625	176.2
Variable	174	16583	87469	89.0
Reduction	47.9%	48.9%	0.17%	49.5%

Stop Criterion

The stop criterion implemented in the SAPPO tool terminates the annealing process when the average solution cost μ exhibits a variation less than $\pm 0.05\%$ during a sequence of w temperature iterations. During the annealing process, the stop criterion is tested only after the range limiter has reached its minimum value. The value assigned to w depends on the speed convergence parameter specified at run-time.

Schedule Parameterization

In the execution of the SAPPO program, different sets of values can be assigned to the parameters defining the inner loop and the stop criterion. This allows for a run-time control of the annealing process convergence speed. Seven different convergence rates, t_i, $i = (1, 2, \cdots, 7)$, can be specified to the algorithm. Each increasing value of i selects a different set of pre-defined values which will slow down the convergence rate in an attempt to increase the quality of the final solution. The set of values listed in Table 5.4 were experimentally tuned to set tighter conditions with increasing i on the acceptance region for the *within count* and on the number of times the average cost variation is expected to remain below the threshold value specified by the stop criterion.

The values in Table 5.4 are used as indicated in equations (5.10) through (5.14) to affect the parameters identified in the inner-loop and stop criteria.

Table 5.4. Parameters controlling the convergence rate.

Rate Index	ξ	k_1	k_2	w
t_1	0.675	1	1/4	3
t_2	0.600	2	1/2	3
t_3	0.500	3	1	4
t_4	0.350	6	2	5
t_5	0.250	9	4	6
t_6	0.150	12	8	7
t_7	0.050	15	16	7

$$\xi = \frac{\delta}{\sigma} \tag{5.10}$$

$$\beta = k_1 \cdot p \cdot \mathrm{erf}\,(\xi) \tag{5.11}$$

$$\gamma = k_1 \cdot p\,(1 - \mathrm{erf}\,(\xi)) \tag{5.12}$$

$$B = k_2 \cdot \frac{p\,(p-1)}{2} \tag{5.13}$$

$$U = 4 \cdot B \tag{5.14}$$

Experimental Results

Several experiments were conducted to assess the stability of the implemented annealing schedule and to quantify the effects of the run-time parameters on the overall behavior of the algorithm. Such an assessment provides guidelines for the application of the tool and validates the produced results.

The first set of tests were designed to quantify the effects of the different convergence rates on the algorithm behavior and final cost. In the experiment, a single circuit containing seven modules was placed multiple times, one per each different convergence rate and the resulting number of trials, temperature behavior, final cost, and number of iterations were recorded. Figure 5.7a shows the growth of the number of attempted moves (trials) as a function of the successive convergence rate factors. It was found that the convergence factors exponentially increased the number of moves required for the algorithm to complete. The quality of the final solution, shown in Figure 5.7b as a function of the number of moves, asymptotically decreased approaching a lower bound as the number of trials increased. This plot suggests that a convergence rate of t_4 or t_5 offers the best accuracy-speed ratio.

The number of temperature steps as a function of the convergence rate factor, illustrated in Figure 5.7c, grew at an approximately logarithmic rate, a growth driven by the narrowing tolerance in the stopping criterion. The values of the initial and final temperatures per convergence rate are shown in Figure 5.7d. The values of the melting temperature, shown at the top of the figure, were not sensitive to the changes in the convergence factors. This

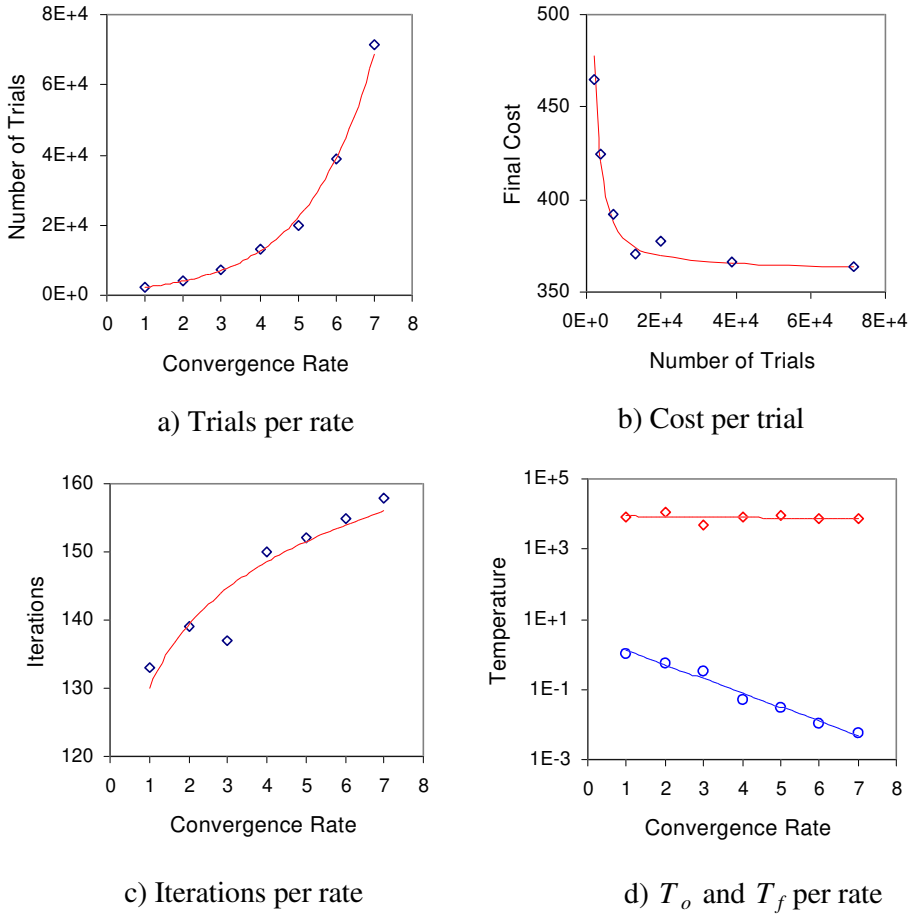


Figure 5.7. Effects of the convergence rates.

was an expected behavior due to the independent formulation given to these parameters. The values of the final temperature decreased at an approximately exponential rate with the growth of the convergence factors. This was also expected from the tighter conditions imposed on the stopping criterion. In general, the behavior of all tested factors resulted as expected, showing consistency in the schedule.

An additional set of tests was performed to assess the stability of the annealing schedule. This assessment can be performed by measuring the spread of the solution cost as a function of the temperature when the same problem is solved several times [94]. Figure 5.8 shows

the outcome of this experiment for three successive solutions of a single circuit with ten macro blocks.

It can be observed that at high temperatures all solutions oscillate in the same neighborhood. As the process converges, the solutions approach the same asymptotic value with very small spread ($\mu = 87464, \sigma = 1323$).

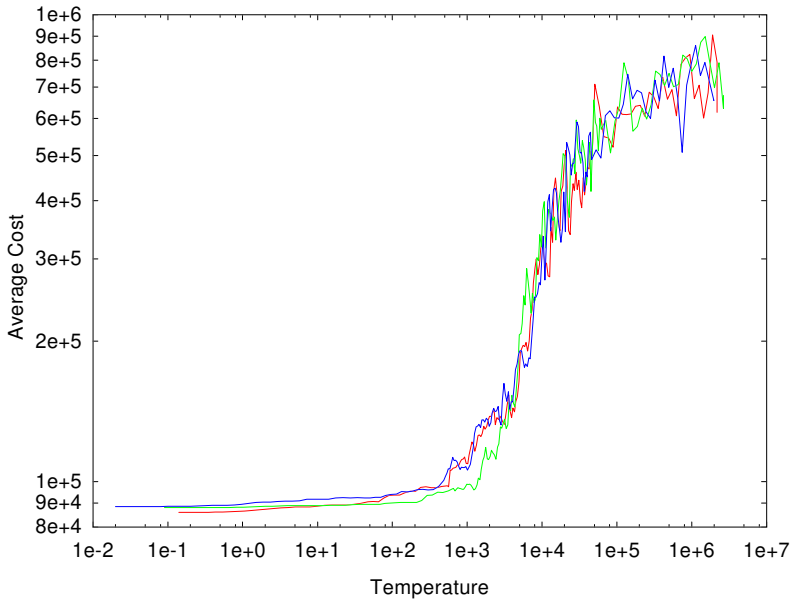


Figure 5.8. Stability of the annealing schedule.

5.4 Summary

This chapter described the implementation details of the SAPPO tool. Details explaining the input-output formats of the program, internal algorithms, and the calibration process of sensitive parameters in the simulated annealing heuristic have been discussed.

Several important achievements were made throughout this process. The development of the input-output formats represent a new physical netlist specification supporting power

parameters. This is an essential step not only for the SAPPO tool, but also towards the development of other mechanisms that could be created to optimize power dissipation during the physical design stage of VLSI circuits.

During the development of the functions used in the cost evaluation of placement solutions, new algorithms emerged allowing the manipulation of complex rectilinear shapes without a need for decomposing them into a set of constituent rectangles. Analyses of the time and space complexities of the algorithms were also included, demonstrating their efficiency. Considering that the solution space size of placement configurations grows exponentially with the number of modules in the layout, these new algorithms avoid the increased space to be dealt with when non-rectangular blocks are decomposed.

The implementation of the annealing heuristic, although conceptually simple, required the calibration of a large number of parameters which turned its practical implementation into a laborious task. A number of small experiments were set up for the calibration process which proved to be effective in their purpose. Although this task did not follow a strictly theoretical procedure, it provides useful guidelines for parameter refinement in similar implementations.

CHAPTER 6

Results and Analysis

This chapter presents the results and analysis of the application of the placement methodology developed as part of this dissertation. A number of experiments performed on a set of circuits selected from the MCNC benchmark suite were used to evaluate and validate the contributions of this research.

The contents of the chapter has been organized as follows. A description of the characteristics of the circuits used for the evaluations and the general approach followed in the experiments are presented in the next section. Section 6.2 discusses the tests performed to validate the heuristic, using as a reference results previously reported on the set of benchmarks. The gains in switched length as well as the effects of using the median-biased heuristic in the placement tool are also discussed in this section. The last two sections contain an analysis of the asymptotic behavior of the methodology and a summary of related ideas.

6.1 Circuit Profiles

The evaluation of the capabilities of the SAPPO tool as a layout placer was carried out on a set of five circuits selected from the MCNC benchmark suite. These circuits, in addition to containing a representative mix of block and net sizes, have been widely used to report comparative results of physical design tools. The characteristics of these circuits, along with some statistics about their contents are summarized in Table 6.1.

Table 6.1. Characteristics of the circuits used in the evaluation of results.

Circuit name	Number of			Average		Standard deviation	
	blocks	nets	pads	block size	net size	block size	net size
ami33	33	123	42	0.035	4.21	0.018	9.23
ami49	49	408	22	0.732	2.34	1.030	1.33
apte	9	97	73	5.173	2.96	1.852	1.93
hp	11	83	45	0.802	3.72	0.664	3.03
xerox	10	203	2	1.935	3.44	1.060	6.01

Two types of analyses were performed on these circuits. The first set of tests was aimed at establishing the quality of the results produced by SAPPO tool with respect to previously reported placement approaches. The second analysis was focused on quantifying the power savings introduced by the switched length approach in the optimization of the circuits. Section 6.2 presents the results of these experiments and their analysis.

A second set of custom configured circuits was used to perform an analysis of the asymptotic behavior of the placement tool. In this experiment, a single circuit containing a mix of blocks of different shapes and sizes was replicated creating several scaled versions of the basic problem posed by the original circuit. Different scaling factors were used in the construction of the replicas and each resulting circuit was placed with the SAPPO tool. The details of this experiment are discussed in Section 6.3.

6.2 Comparative Analysis

The comparative analysis of the results produced by the SAPPO tool was divided into two stages, the first to establish the quality of results produced by the tool and the second to quantify the power savings attained through its use.

To validate the results produced by the SAPPO tool, the cost function in the simulated annealing algorithm was customized for producing placements optimized for wirelength only. This enabled the collection of a set of results that could be compared with those from previous approaches reporting wirelength results on the target circuits. Once this reference

was established, the target circuits were placed again, this time with the original objective function. The new set of results were then compared for switched length with those from the previous placements.

6.2.1 Wirelength Results

The formulation given to the cost function in the annealing algorithm allowed for easily retargeting the placement objective by changing the terms making up equation (5.1) to only reflect wirelength. The resulting expressions for these terms, denoted $C_1'(S)$, and $C_2'(S)$ were

$$C_1'(S) = \sum_{j \in N_1} L_j + \sum_{k \in N_2} B(n_k) \tag{6.1}$$

$$C_2'(S) = \psi' \cdot \sqrt{(TO(S) + AO(S)) \cdot C_1'(S)}, \tag{6.2}$$

where L_j, N_1, N_2 and $B(n_k)$ are defined as in Section 5.3.1. A new scaling constant $\psi' = \frac{1}{2\sqrt{2}}$ was required to properly balance the weights of the wirelength and overlap in the cost function.

The wirelength measurements obtained with the customized objective function are summarized in Table 6.2. along with similar results reported in the literature for a placement heuristic with a similar objective [113]. The results from the SAPPO program were obtained for a convergence factor t_5 and using the semi-perimeter method as the wirelength estimator. This last condition was introduced to use the same estimator as in most of the compared techniques. All wirelength measurements are given in mm and areas are expressed in mm^2. Table 6.3 shows additional results reported for the two most frequently reported circuits.

The outcome of this experiment shows that the placements produced by the SAPPO tool are, in terms of wirelength and area, competitive with the best results published for the target circuits. It should be considered that although the same parameters are reported on the same set of circuits, a direct comparison among the techniques in the above tables

Table 6.2. Wirelength optimization for the SAPPO tool and a reference heuristic.

Circuit	SAPPO		BB [113]	
	Wirelength	Area	Wirelength	Area
ami33	77.18	2.50	109.0	2.24
ami49	835.03	60.14	1021.0	51.49
apte	366.10	54.69	460.0	54.05
hp	223.59	15.36	278.0	12.15
xerox	613.37	29.71	628.0	26.60

Table 6.3. Additional results reported on the set of benchmark circuits.

Circuit	SAPPO		Bear [121]		MBP [57]		TimberWolf [99]	
	Wire	Area	Wire	Area	Wire	Area	Wire	Area
ami33	77.18	2.50	131	2.83	91	2.42	105	2.57
xerox	613.37	29.71	633	28.47	601	25.79	-	-

might be difficult to establish. This is because each of the referred tools operates under different conditions. Examples of these differences include the fact that all the reference tools optimize area while the SAPPO considers it just as a constraint. Also, some of the tools report their results after routing has been completed on the circuits [57], [121], and some even include a stage of compaction [99]. Regardless of these differences, the results from the SAPPO tool can still be validated through these tables.

An additional item of information regards the CPU time requirements. Although it is practically impossible to make a meaningful comparison of execution times for the above set of placement mechanisms, CPU time is reported to provide some idea of time requirements. Table 6.4 indicates the amount of CPU time taken by the SAPPO tool to produce the reported results. The tool was written in C and the reported execution times were obtained on a Sun Sparc Ultra 10. The times taken by BB to produce the solutions above for circuits `xerox` and `ami33` on a DEC3100 were 66.7 minutes and 89.1 minutes, respectively.

Table 6.4. Running times for the SAPPO program.

Circuit	ami33	ami49	apte	hp	xerox
Time (min.)	50.94	332.86	8.07	7.77	8.17

6.2.2 Switched Length Results

To produce results where an assessment of the power dissipated by the wires of the layout could be made, a new set of placements were generated using the original cost function $C(S)$ which optimizes switched length. Four sets of data were recorded from these placements: global wirelength (WL), total switched length (SL), length of the longest net (DCN), and the cost of switched length critical net (PCN). Similar sets of parameters were also recorded in the previous placements optimized for wirelength whose results were discussed in Section 6.2.1.

The goal of collecting the data on the above four parameters was to quantify the savings in power that could be obtained by targeting the cost function to reduce the circuit switched length rather than its wirelength and/or area. In these experiments, the core area was established as a fixed constraint on all placements and no constraints at all were placed on either the maximum DCN or PCN. The reason for tracking these last two parameters, even when they had a free behavior in the processes, was to quantify the effect of the new placement technique on the critical delay path and the power critical net of each circuit. Table 6.5 contains the results obtained with the wirelength cost function $C'(S)$ including the additional recorded parameters, along with the results from the switched length cost function.

As before, all lengths including switched length, are expressed in mm. A summary of the improvements obtained by the application of the switched length cost function on the recorded parameters is provided in Table 6.6.

The results in Tables 6.5 and 6.6 indicate that targeting the objective function to reduce the switched length of the circuits consistently produced modest improvements in the power

Table 6.5. Wirelength results versus switched length on the set of benchmarks.

Circuit	Cost function: $C'(S)$				Cost function: $C(S)$			
	WL	SL	DCN	PCN	WL	SL	DCN	PCN
ami33	77.18	16.67	2.70	1.29	78.31	14.48	2.74	1.20
ami49	835.03	206.34	12.07	5.04	884.74	205.76	13.31	5.06
apte	366.10	76.62	12.47	4.85	353.26	69.10	11.63	4.99
hp	223.59	51.85	7.49	2.77	223.57	50.57	7.83	2.90
xerox	613.37	137.48	10.10	3.45	537.75	115.15	9.44	3.02

Table 6.6. Improvements obtained when switched length was used as target.

Circuit	WL (%)	SL (%)	DCN (%)	PCN (%)
ami33	−1.46	13.14	−1.67	7.19
ami49	−5.95	0.28	−10.33	−0.29
apte	3.51	9.82	6.74	−2.89
hp	0.01	2.46	−4.61	−4.43
xerox	12.33	16.25	6.56	12.45

dissipation of the layouts. An average reduction of about 7% over all circuits was achieved, with most of them showing improvements of 10% or more. At the same time, these savings in power did not degrade the already established good results in wirelength for most of the circuits. An example of an output layout is illustrated in Figure 6.1.

A small degradation, however, was observed in the length of the DCN for several circuits. This behavior can be attributed to the redistribution of weights induced by the switching activities on the circuit networks. It was observed that in most cases the nets flagged as DCN were among the least active of the layout. The average degradation though was very small: -0.3% over the entire set of circuits. The power critical net, although showing a small average improvement of 1.4%, also had an oscillatory behavior. The results in the last two parameters indicated that meaningful behaviors could only be established if the cost function was modified to optimize critical power dissipation along with the overall switched length.

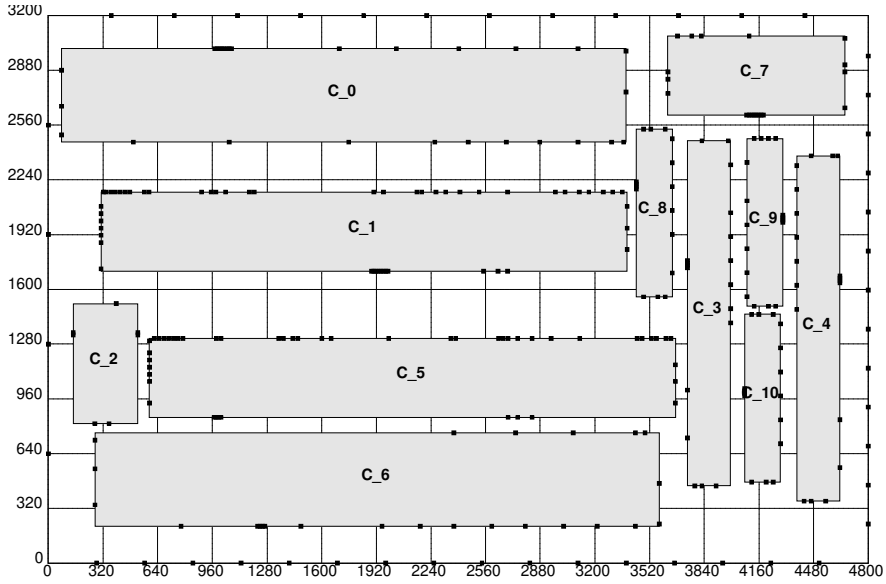


Figure 6.1. Placement of benchmark `hp`.

6.2.3 Other Results

Additional tests were performed to quantify the effects of the wirelength estimator on the SAPPO results. Two estimators were considered: the MBH, discussed in Chapter 4, and the semi-perimeter method (SPM), described in Chapter 2. The interest on the SPM originates from the fact that it is used by virtually all reviewed placement techniques, including those used for comparisons in the previous section.

The specific aspects investigated were the impact on the quality of the final solution and the effect of the estimator on the run time of the placement heuristic. Three circuits were used in the tests, `ami33`, `apte` and `hp`, each of which was optimized for switched length under the MBH and compared to the previous results already based on the SPM. Prior to making any comparisons, the final wirelength of the circuits placed under the SPM was re-estimated with the MBH. This conversion was necessary to fairly compare both measurements under the same metric.

The analysis on the converted data revealed that despite the underestimation trend known to affect SPM measurements in circuits with regularly shaped modules like `ami33`

and `apte`, the improvements induced by the MBH on the final switched length were minimal (less than 3%). For less regular blocks like those in benchmark `hp`, the MBH produced a final switched length 13% smaller. More consistent values were observed in the DCN and PCN where the MBH produced values reduced in up to 35% and 38% respectively, and never less than 11%.

The above improvements were outweighed, however, by the excessive time requirements of the MBH. To illustrate the disparity in CPU times, the placement of benchmark `hp` with a convergence speed factor of t_3 was completed under the SPM in 107 seconds. Under the same conditions, the MBH required 6117 seconds.

It should be recalled from Chapter 4 that although the MBH was proven competitive with other heuristics in its class, it has a time complexity $O\left(n^2 \cdot \log n\right)$ on a graph with n vertices. In the implementation of the SAPPO algorithm, this is added to the overhead of the graph construction and reduction mechanisms, bounded by an $O\left(n^2\right)$ polynomial. In contrast, the SPM has a time complexity $O\left(n\right)$ for a network with n terminals.

6.3 Asymptotic Behavior

An issue seldom discussed in most heuristic techniques is the method used for evaluating their effectiveness. When a deterministic algorithm is applied to a problem where the desired optimum value can be somehow estimated or measured, the evaluation of the algorithm effectiveness becomes a trivial task. However, when *NP*-hard problems like those found in VLSI layout are approached, their optimum solutions are usually intractable and therefore comparative evaluations require careful handling. This fact becomes more critical when the underlying optimization mechanism is of a stochastic nature like in the case of simulated annealing.

The subject of assessing the effectiveness of such a class of heuristics has been addressed by Hagen, *et al.* [122]. Their approach establishes that given a solution cost $C_H(I)$ obtained by executing heuristic H on problem instance I, then a k-scaled instance of I for some $k > 0$ will yield a solution $C_H\left(k \cdot I\right) \geq k \cdot C_H\left(I\right)$, when solved using H. Then, the *scaled*

suboptimality of heuristic H for a scale factor k, $\eta_H(k)$, can be computed as

$$\eta_H(k) = \frac{c_H(kI)}{kc_H(I)} - 1. \tag{6.3}$$

A heuristic will be considered sub-optimal if $\eta_H(k) > 0$. The growth of this factor can be used to predict how bad a heuristic can become as the problem size increases.

Tests were performed to measure the asymptotic behavior of the SAPPO heuristic using the scaled suboptimality criterion explained above. In these tests, a single circuit consisting of seven macro blocks was replicated 2, 4, and 8 times to create scaled versions of the initial layout, as illustrated in Figure 6.2.

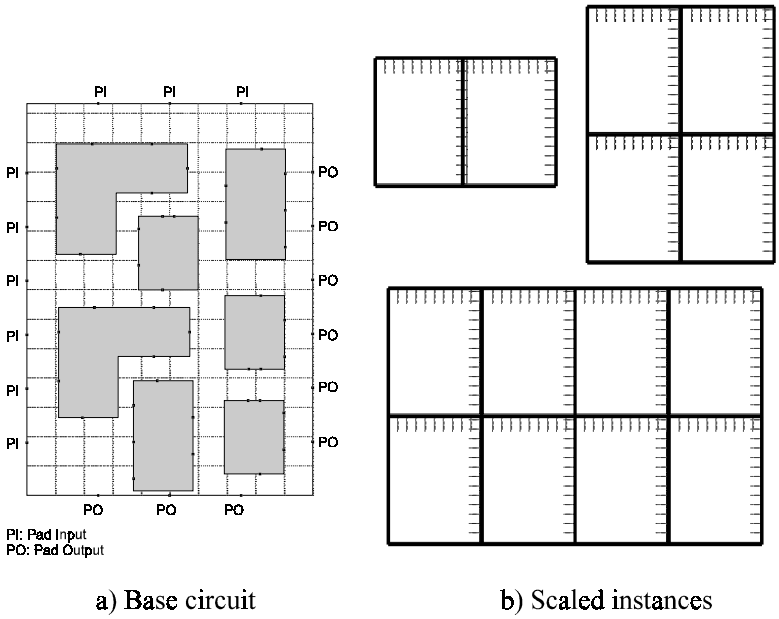


Figure 6.2. Base and scaled circuits used for asymptotic tests.

Given that the heuristic to be tested works on a per-net basis, the scaling factor k for each new instance was computed based on the increasing number of nets of the new circuits. Table 6.7 summarizes the characteristics of the original and scaled layouts.

Table 6.7. Characteristics of scaled layouts used for asymptotic behavior tests.

Circuit	Blocks	Nets	I/O	Area $\times 10^3$	k
ex7	7	24	18	19.2	1.00
ex8	14	42	24	38.4	1.75
ex9	28	78	36	76.8	3.25
ex10	56	144	48	153.6	6.00

Multiple runs of each circuit were carried out and the resulting cost, reflecting the layout switched length, was averaged to compute the scalability factor η for each instance. The results of the experiment are summarized in Table 6.8.

Table 6.8. Results of suboptimality tests.

Circuit	Runs	SL	η	Deviation
ex7	5	534	0.0	—
ex8	3	937	$2.68e-3$	0.27%
ex9	3	1806	$40.62e-3$	4.06%
ex10	2	3427	$69.60e-3$	6.96%

The results of the scalability tests indicate a satisfactory behavior of the heuristic, as can be appreciated from the small values of η. In the largest of the test cases, the deviation from the optimal scaling behavior was less than 7% and, within the obtained values, the overall behavior experiences a near-logarithmic growth.

6.4 Summary

The results obtained from the application of the placement methodology developed as part of this dissertation were presented and discussed in this chapter. A comparative analysis of the wirelength quality of the layouts it produced ranked its results among the best known for a selected set of benchmark circuits. The same set of high quality results were then

evaluated for their power dissipation characteristics and compared to those obtained when the newly developed methodology was used to reduce the switched length of the circuits. It was found that through its application, modest reductions in the power dissipation of the test circuits were achieved without degrading the quality of the placements. Moreover, an analysis of the asymptotic behavior of the heuristic pointed to a good scalability of its results as the size of problems increase.

Additional results were analyzed showing that further improvements were possible by replacing the semi-perimeter-based wirelength estimation mechanism used in most tests by an estimator based on the median-biased heuristic. However, the improvements obtained in switched length and critical nets are outweighed by the time requirements of this alternate method. It is inferred by this outcome that such a heuristic might be more appropriate for the stages of global and detailed routing where its good stability and consistency might counterbalance its time complexity.

CHAPTER 7

Conclusion

The design, implementation, and evaluation of a power reducing placement methodology for macro block-based VLSI layouts has been presented in this dissertation. The results of its application have been presented and analyzed indicating that power reductions are possible without significantly degrading the quality of the output layouts in terms of area or delay.

This chapter summarizes the work developed in this dissertation, highlighting the contributions that have emerged through its completion, and identifying future research directions.

7.1 Research Summary

The review presented in Chapter 2 exposed the relationships among circuit parameters and input data patterns in the power dissipation of digital circuits. This analysis identified the supply voltage, the switching activity, and the network capacitance as the variables defining the solution space for low-power tools. An analysis of the problem of estimating switching activity in CMOS circuits was also presented, discussing salient techniques, identifying the different estimation trends, and highlighting their strengths and limitations. This analysis supported the selection of a probabilistic switching activity estimator for the power driven placement environment.

The study of power minimization techniques made it evident that although numerous

approaches have been proposed with different degrees of success, the physical design stage has received the least attention. Moreover, a review of placement techniques indicated that although this has been the most intensively researched physical design problem, virtually all reviewed techniques have neglected the effects of placement configurations on the power dissipation of digital circuits. These facts made evident the existence of a gap in the development of power conscious physical design techniques.

The problem statement in Chapter 3 describes the circuit placement for minimum power dissipation. This statement, in addition to formally presenting the problem, made evident the complexity involved in the search of a solution. An important point stressed in this chapter was the difference between wirelength and switched length optimization.

In Chapter 4, error skew effects on techniques used for estimating minimum interconnection networks in a power optimizing physical design environment was studied. Considering the speed-accuracy constraints of the application framework, techniques falling in the class of greedy Steiner minimum tree (SMT) estimators were considered. The study found that the error behavior of typical estimators could have a drastic effect on the indicated application framework. Two new heuristics, the MBH and the DBH were proposed and evaluated for their error behavior. Results showed that among several SMT heuristics, the MBH had a better consistency of quality results for large nets, a desired characteristic in the application framework of the estimators.

To limit the search space of RSMT heuristics, selected graph reduction techniques were adapted to the requirements of layout graphs. They were found to be both time efficient and powerful. Results indicate average deletions of over 95% of Steiner candidates in more than 90% of the cases.

The implementation of the placement tool supporting the proposed power reduction mechanisms was presented in Chapter 5. Details explaining the input-output formats of the program, internal algorithms, and the calibration process of sensitive parameters in the simulated annealing algorithm were discussed. Among important achievements of this process were the development of a new physical netlist format supporting power parameters

and new algorithms proposed for the manipulation of complex rectilinear shapes as single objects. Analyses of the time and space complexities of these algorithms were performed demonstrating their efficiency. In the implementation of the simulated annealing algorithm, a careful experimental calibration process was described for the annealing schedule, the cost function, and the perturbation mechanism. This calibration led to a very stable and consistent adaptive implementation of the annealing algorithm.

The main results of this dissertation were presented and evaluated in Chapter 6. A comparative analysis of wirelength and area on a set of benchmark circuits placed for minimum wirelength was used to validate the results produced by the placement tool. Comparing these to the placements produced with the developed low-power methodology, it was found that modest reductions in the switched length of the test circuits were achieved without degradations in wirelength or delay. In about 60% of the tested cases, power reductions in signal nets from around 10% to over 16% were measured. Moreover, an analysis of the asymptotic behavior of the heuristic pointed to a good scalability of these results as the size of problems increase.

Additional results were analyzed showing that further improvements were possible by replacing the semi-perimeter-based wirelength estimation mechanism with an estimator based on the MBH. Despite the high cost in execution time at which these additional improvements were obtained, this analysis confirmed the hypothesis of the effect of the skew error of estimators in power-driven physical design applications. Of special interest were the effects caused by this approach on the critical nets of the circuit, where peak reductions in excess of 35% were measured.

7.2 Contributions

The objectives of this dissertation required an approach that combined diverse areas of knowledge. At the same time, the development of this work produced several contributions which enhance several of these areas. Five major contributions can be attributed to this research. Below, a summary is presented of these new concepts.

A New Low-power Placement Methodology: The methodology developed as part of this dissertation was shown to be an effective way of reducing the power dissipation of new physical layouts. The results obtained indicate that stable and consistent reductions in the switched length of the output layouts can be obtained with very small degradations in wirelength and delay. Although this was an experimental program, the concept has been proven to work. This implies it could be used either in emerging industrial grade low-power design tools or incorporated into existing programs.

A Better Understanding of Power Dissipation Patterns in Physical Layouts: The results obtained supporting the newly developed methodology have provided a better insight into the power dissipation patterns of macro-block based layouts. Researchers have conjectured that it might be possible to control the power dissipation of circuits by introducing changes in their layout design methodology [123]. However, no research has been found reporting on this behavior for the wiring of macro block-based layouts. The experiments performed with the power optimizing placement tool have quantified these gains for the indicated layout style. Moreover, these results show how and at what cost diverse degrees of reduction can be achieved. An additional item in the quantification of power savings achievable during physical layout design is that for the studied style, the obtained results point to the placement stage as that producing the highest power reductions.

An aspect for which no previous discussion has been found is the effect of the skew error of network interconnect estimators on the resulting power dissipation of layout designs. The results of this research have pointed out that nets with a large number of terminals typically become critical in the power dissipation. However, these nets are frequently the least accurately modeled by traditional interconnect estimators. This research quantified part of the effect caused by wirelength estimators in the total switched length during the placement stage. Evidence was presented showing that there could be a more dramatic impact in the results of performance driven placement techniques. It is conjectured at this point that the same reasoning could be applied to a hypothetical router for low-power, or a performance driven routing application.

New Heuristics for the Minimum Rectilinear Steiner Tree Problem: Two new heuristics for the minimum rectilinear Steiner tree (MRST) problem in graphs were developed. These new algorithms were found to be competitive in terms of space-time complexity and error bound with other heuristics in their class. Most important, the MBH was found to outperform similar heuristics when presented with configurations of increasing numbers of terminals. This result might prove useful in many areas where rectilinear Steiner trees find application. This is especially true for the development of power reducing layout techniques, as was demonstrated in the wirelength estimation application in low-power placement.

An additional contribution relevant to the MRST problem was the development of an efficient method to perfom reductions on graphs produced from two-dimensional spaces with obstructions. A set of mechanisms previously reported for graphs extracted from obstacle-free layouts were selected and adapted for application in graphs generated from macro block-based layouts. The selected subset was implemented and found to be as effective as other mechanisms having larger time complexity.

Improved Algorithms for the Manipulation of General Rectilinear Shapes: A set of algorithms for the manipulation of general rectilinear shapes capable of handling non-rectangular objects as single entities were developed. These algorithms were found to have space and average time complexity similar to that of mechanisms which handle non-rectangular shapes as a set constituent rectangles. The relevance of this contribution resides in the reduced number of objects needed to be considered when handling complex rectilinear shapes. Consider as an example the case of an iterative improvement placement application. When non-rectangular shapes need to be decomposed into constituent rectangles the size of the solution space explored by the application dramatically increases. This is because of the exponential growth of the solution space with the number of modules.

An Extended Netlist Format Supporting Low-power Layout Techniques: Extensions were created to a standard physical netlist format expanding its capabilities to support power optimizing physical design tools. With this contribution, part of the foun-

dation for the development of a unified set of low-power tools covering all physical layout design subproblems has been established.

7.3 Future Work

The creation of a power optimizing placement tool for macro block-based layouts, as was indicated earlier, only partially covers the need for power conscious physical design tools. Similar approaches are needed in the remaining stages of layout design, and even in the placement stage itself, to provide a continuum of tools enabling power optimized layouts from beginning to end. This necessity points to multiple development objectives for tools that could partition, route, and compact physical layouts. The development of such tools could take advantage of several mechanisms already developed for the placement tool.

Another interesting development that would enhance the low-power placement methodology would be its expansion to handle layout styles other than macro blocks. Given the high degree of placement freedom offered by the macro block style, the introduction of appropriate constraints could adapt the algorithms to handle standard cells, gate arrays, or even mixed layouts. In all of these cases suitable rules for low power optimization could be devised using the switched length of nets as a measure of quality in the resulting layouts.

In the analysis of the effects of wirelength estimation accuracy, results consistently showed large differences in the costs of critical nets. A more detailed analysis of this trend in performance-driven physical design techniques could provide an insight into a problem that might not have been noticed before.

Despite the quality of the results reported from the application of the developed placement tool, the current implementation has not been optimized for speed. Several options have been identified as possible sources of improvement.

The most time consuming tasks during the MBH-based operation are the wirelength estimation and the overlap computation. An alternative that could be explored for speeding-up these processes would be through their parallelization. Some data dependencies exist between the processes, but given that overlap computation is based on modular information

and wirelengths on network topologies, the data in the two stages have a large degree of coarse grain parallelism. The small dependency arises in the detection of feasible nets, made with the use of the segment trees of the overlap detection mechanism.

Another alternative worth exploring for speeding-up the process would be partitioning the layout into subcircuits and parallelizing their optimization. In this case, processor interaction would occur due to the presence of nets joining the sub-circuits. To minimize this interdependence in the data, a partitioning algorithm that minimizes the cut size could be used. Issues needing additional consideration include the possible performance degradation that might occur due to the fragmentation of the solution space.

7.4 Closing Remarks

The ideal behind this dissertation, more than simply providing an isolated method for power efficient placement of a certain type of layouts, is the creation of a unified design methodology that could transform the way in which physical design automation tools operate from beginning to end.

This is not an easy goal. After reaching this point and reflecting on all the work it took to complete this dissertation, one realizes that this is just a small step in reaching that ultimate objective. The foundation have been laid and the objectives are clear.

APPENDICES

APPENDIX A

Using the SAPPO program

This appendix briefly describes the use of the SAPPO program. It includes details of command line options used in its invocation, the input format specification and the different output files it produces. Also, summarized information is provided on a translation tool developed to facilitate the task of importing layouts from other languages.

A.1 Command Line Syntax

The SAPPO program can be invoked by its name from the command prompt. If no command line parameters are specified, the program produces the following message on the standard output device:

```
SAPPO: Simulated Annealing Placer for Power Optimization.
Usage:
sappo [options] <filename>

filename = name of file with netlist in Low-Power YAL (LPY) format.

OPTION  EFFECT
  -f   Produce a file.FIG with a plot of placement in xfig 3.1 format.
  -l   Produces a log file file.LOG of the program execution.
  -o   Produces an LPY file with the resulting layout.
  -c   Evaluates the cost of a solution, bypassing the annealing process.
  -t#  Annealing time control. Value #: 7=slow, 1=fast, default=3
  -r # Specifies a seed (# > 0) for random number generation, default=auto.
  -i   Generates an INITIAL solution different from that in the input file.
  -w   Annealing process to optimize Wire length only.
```

```
Steiner approximation obtained through:
 -m   Median biased path heuristic (MBH) --default
 -d   Distance-network heuristic (DNH)
 -p   Prim's Minimum Spanning Tree (MST)
 -s   Semi-perimeter method (SPM)
```

The different command line arguments can be combined to specify concurrent actions in the processing of the circuit. Only one Steiner approximation method can be specified at a time. The program will use the provided file name to search for an LPY circuit description file and a supplemental rules and constraints file with the same name as the input circuit and extension RCF. The formats of these two files are described in the sections below.

A.2 Input Format

The layout input/output format was developed based on YAL layout language, originally developed by B. Preas and K. Robers[*] and used in the MCNC benchmark circuits. It also includes modifications introduced later by J. Rose[†]

The new format, named LPY (low-power YAL), extends the YAL constructs to support the specification of power dissipation parameters in physical design netlists. The description of the LPY format presented below closely follows that of the YAL language. It includes the newly added constructs that extend its support to physical design for low power.

The new features added to YAL include an additional section in the layout parent module specifying the switching activity factors of input and internal nodes of the circuit. Terminal descriptions in all module types have been extended to allow the specification of their equivalent resistance and capacitance values, facilitating the computation of all power components of the layout as well as extending the previous support for delay computations.

The description of critical nets has been complemented with a new section added to specify module propagation times and timing constraints for signals traveling through the entire chip. Module dimensions in the LPY format remain specified by the coordinates

[*]YAL is a Copyright (C) 1987 of Bryan Preas and Ken Roberts. All rights reserved, March 1987.

[†]Modified by Jonathan Rose to accomodate rectilinear cells, power and ground nets with current and voltage restrictions, and the specification of critical net lengths. The new YAL is not compatible with the old. November 1987.

of the corners of their rectilinear shapes in counter-clockwise order, as in its predecessor. However, the LPY format enforces that the leftmost of the lower corners is specified first.

In the YAL format, design rules are not part of the language. This remains true for the LPY, but a separate file is now provided which allows for specifying a reduced set of design rules and constraints.

A.2.1 The LPY Netlist Format

An LPY file is composed essentially of comments and data lines. Comments are delimited by "/*" and "*/" and may extend across physical line boundaries. Data lines are composed of a series of keywords and tokens organized to form diverse sections in the format. White spaces are used delimiters between tokens. A white space can be one or more of any combination of space, tab, line feed, or carriage return characters.

The data is free format. A semicolon, ";", is the logical line terminator. Any number of white space characters may separate a semicolon. A logical line can occupy more than one physical line. Any number of logical lines can occupy a physical line.

Within the format, "name" fields are used for module or cell definitions, module or cell instances, and signals or nets. Names are case sensitive, may contain any combination of characters except white spaces or semicolons, and do not require to begin with an alphabetic character.In the descriptions below, "|" means OR, implying the selection of one item from the list of fileds separated by the "|". Items enclosed in square brackets, "[" and "]", indicate optional fields. An ellipsis, ". . .", either horizontal or vertical indicates that the preceding field or line can be repeated as necessary. Tokens enclosed by angle brackets, "<" and ">", are used as intermediate definitions. Words in upper case are keywords of the language. A "number" is a real value and can specify a measure in microns or the value of some other quantity.

Module Definitions

A module is a definition of a circuit being laid out or a constituent primitive cell. The definition for each circuit or primitive cell begins with "MODULE <modulename>". The

definition ends with the "ENDMODULE" keyword. A template of a module description is presented below.

```
MODULE <modulename>;
     TYPE <moduletype>;
     DIMENSIONS <X1> <Y1> <X2> <Y2> . . . <XN> <YN>;
     IOLIST;
          <signalname> <terminaltype> [<Xposition> | <side>
          [ <Yposition> | <position>  [<width> <layer>]]]
          [CURRENT <current>] [VOLTAGE <voltage>]
          [RESIS <resistance> CAPAC <capacitance>];
          .
          .
          .
     ENDIOLIST;
     NETWORK;
          <instancename> <modulename> <signalname> . . . ;
          .
          .
          .
     ENDNETWORK;
     ACTIVITY;
          <signalname> <activity> . . . ;
          .
          .
          .
     ENDACTIVITY;
     PLACEMENT;
          <instancename> <xlocation> <ylocation>
               [<reflection>] [<rotation>];
          .
          .
          .
     ENDPLACEMENT;
     CRITICALNETS;
          <signalname> <maximumlength>;
          .
          .
          .
     ENDCRITICALNETS;
     PIN2PINTIME;
          <signalname> <signalname> <risetime> <falltime>;
          .
          .
          .
     ENDPIN2PINTIME;
```

```
ENDMODULE;
```

The different fields in the module template described above are defined as follows:

```
<modulename> ::= name
<moduletype> ::= STANDARD | PAD | GENERAL | PARENT | FEEDTHROUGH
<width> ::= number
<height> ::= number
<signalname> ::= name
<terminaltype> ::= I | O | B | PI | PO | PB | F | PWR |  GND
<side> ::= BOTTOM | RIGHT | TOP | LEFT
<layer> ::= PDIFF | NDIFF | POLY | METAL1 | METAL2
<xposition> ::= number
<yposition> ::= number
<position> ::= number
<instancename> ::= name
<xlocation> ::= number
<ylocation> ::= number
<Xi> ::= number
<Yi> ::= number
<current> ::= number
<voltage> ::= number
<activity> ::= number
<resistance> ::= number
<capacitance> ::= number
<maximumdelay> ::= number
<risetime> ::= number
<falltime> ::= number
<reflection> ::= RFLNONE | RFLY
<rotation> ::= ROT0 | ROT90 | ROT180 | ROT270
```

The "TYPE" line is required for all modules. Allowable module types are: PAD for a driver or pad cell that should be placed around the periphery of the chip. GENERAL for an internal, primitive general cell or building block. PARENT for a higher level module to be laid out. Additional types supporting standard cell-based layouts are: STANDARD for an internal, primitive standard cell that must be placed on a row or column, and FEEDTHROUGH: a cell used to connect a signal across a row.

Whether or not a line like DIMENSIONS or a section like NETWORK or PLACEMENT is required within a module depends on the TYPE of module being defined. A TYPE

and an IOLIST are always required. A primitive cell is assumed to have been laid out previously and therefore has the following data: DIMENSIONS and IOLIST; <xposition> or <side>, <yposition> or <position>. Tokens <width> and <layer> are required for each of the IO <signalname>s within the IOLIST for primitive cells. NETWORK, ACTIVITY, and PLACEMENT sections are ignored for primitive cells. A feedthrough cell (TYPE = FEEDTHROUGH) should not appear in a NETWORK but is inserted into a layout by the layout system as necessary.

A module that defines a circuit to be laid out has TYPE PARENT and must have IOLIST, NETWORK, and ACTIVITY sections. Its DIMENSIONS line can be overridden by information in the file of rules and constraints. The DIMENSIONS line gives the coordinates of the corners of the rectilinear box in counter-clockwise order, with the leftmost of the lowest corners being first.

The IOLIST section defines the external connections or terminals of the module and should be included in definitions of all modules. Each line in this section defines a segment of connection to a higher level of hierarchy for the module. Each line must contain at least a <signalname> of an IO terminal and its <terminaltype>. Within a module being laid out (TYPE = PARENT) a <signalname> on a terminal line in the IOLIST is in the same name space as the <signalname>s in the NETWORK of that same <modulename>. Each module has its own separate name space. Other fields may be required depending on the terminal and the module that contains it as described below.

If a primitive module (TYPE = STANDARD, PAD, GENERAL, or FEEDTHROUGH) is being defined, the lines within an IOLIST must contain the <xposition>, <yposition>, <width>, and <layer>. Terminals appear in the order in which they will be referenced in the NETWORK section of any higher level module. Electrically equivalent terminals (*e.g.*, in a dual-ported primitive module) must have the same <signalname>s. In this case, binding between the terminal <signalname>s of a primitive module and the <signalname>s of the higher level module are performed in the order of unique signal names. It should be assumed that electrically equivalent terminals can be used as "feedthroughs" for the signals

connected to them. If additional over-cell routing is allowed, *i.e.*, for signals not connected to the cell, terminal locations for these feedthoughs will appear at the end of the IO terminal list. Valid <terminaltype>s are:"I" for input. "O" for output. "B" for bidirectional. "Px" for a pad terminal on a primitive, with x is replaced by either I, O, B for input, output, and bidirectional pads, respectively. "F" for an independent feedththrough."PWR" for VDD power input. "GND" for VSS ground input.

All signal terminals can optionally have their equivalent resistance and capacitance values specified to allow for delay computations in timing constrained designs. Resistance values are given in milliohms and capacitance values in picofarads.

Terminal locations are specified by <xposition> (or <side>), <yposition> (or <position>), <width>, and <layer> (or LEFT). <xposition> and <yposition> give the coordinate of the center of the terminal in microns on the edge of the cell. To be compatible with the old YAL, terminals can also be specified as <side> and position. <side> may be BOTTOM, RIGHT, TOP or LEFT. The <position> token gives the coordinate of the center of the terminal in microns along the indicated <side> from the left or bottom of the module as appropriate. Field <width> specifies the dimension of the terminal along the indicated edge; it extends from <xposition> - <width>/2 to <xposition> + <width>/2 or <yposition> - <width>/2 to <yposition> + <width>/2 as appropriate, and <layer> specifies the conductor layer of the terminal.

A fixed pad placement for higher level circuits (TYPE = PARENT) requires a slightly different interpretation of the fields and is specified in the following manner. In this case, the <side> and <position> specifiers must be used, rather than the <xposition> <yposition>. If the terminal is bound to the pad of an IO driver, then the <side> refers to the side of the chip and <position> refers to the position of the origin of that driver cell. <width> and <layer> are ignored. If no <position> is specified, then the pad is restricted to the indicated <side> but not to a <position> on the <side>. If no <side> is specified for a terminal bound to an IO driver then that IO driver may be placed on any <side>.

The NETWORK section defines the internal connectivity for the module. If a NET-

WORK section is included in the definition of a primitive module (TYPE = STANDARD, PAD, GENERAL, or FEEDTHROUGH), it will be ignored. Each line defines an instance of a primitive module that is included in the circuit. Each instance definition has the following fields: <instancename> is the name of the instance of a module. An instance has signal bindings, a location, and an orientation to be determined by the layout system in the PLACEMENT section. <modulename> is the name of the module to which the instance is bound. <signalname> is a list of <signalname>s connected to the module, in an order determined by the module definition.

Connections to the feedthrough terminals are not included in this list; they should be determined by the layout system. Single terminal <signalname>s are used to specify an unconnected terminal. An unconnected terminal may also result if the list of <signalname>s on an instance definition is shorter than the list of unique <signalname>s in the IOLIST of the primitive module being bound.

The estimated switching activity values of circuit nodes are specified in the ACTIVITY section. This new section is expected to be found in the bounding module of the layout (TYPE = PARENT). Each line in the ACTIVITY section contains two fields: <signalname> and <activity>. The <signalname> field defines a unique signal name of those included in the network section. Entries are expected to have the following order: terminals listed in the IOLIST section of the PARENT module, keeping their original relative order, followed by internal net definitions on the NETWORK section, in their relative order of appearance. The second field, <activity> specifies the estimated switching activity of the node identified by <signalname>.

The PLACEMENT section is used input initial placements or to report placement results, one logical line per instance. If a PLACEMENT section is included in the definition of a primitive module (TYPE = STANDARD, PAD, GENERAL, or FEEDTHROUGH), it will be ignored. The fields in a placement section are defined as follows: <instancename> is the cell instance, and must match an instance name in the NETWORK section. <xlocation> <ylocation> specify the location of corner (0, 0) of a module after any orientation change

of the instance. <reflection> <rotation> are the optional reflection and rotation of an instance. The default for <reflection> is RFLNONE (the normal or "defined" reflection), while RFLY means reflected about the Y axis. The default value for <rotation> is ROT0 (no rotation); ROT90, ROT180, ROT270 indicate a counter-clockwise rotation for the appropriate number of degrees. Reflection is applied before rotation.

The dimensions in all primitive cells are forced to the lower left corner of the layout (0,0). If the circuit contains an initial placement specification it must be given through the placement section of the format.

The CRITICALNETS section is used to specify which nets are performance-critical, and their maximum allowable lengths. Field <signalname> identifies the critical net, and must match a <signalname> in the NETWORK section. Value <maximumlength> is a number in microns specifying how long the signal wire can be restricted to. This information is useful when performing performance constrained placement.

A new section, PIN2PINTIME, has been added which allows for specifying pin-to-pin propagation times. Two <signalname> fields specify a pair of terminals in the module of concern. These fields must match <signalname> declarations in the IOLIST section of the module. Fields <risetime> and <falltime> specify the estimated times taken by rising and falling signals, respectively, propagating between the pair of specified terminals. Note that for modules other than PARENT, the pin-to-pin propagation times are input data, while for PARENT modules they represent timing constraints for critical signals traveling through the chip.

A Macro Block-based Example

The description below corresponds to a layout consisting of three primitive macro cells, each instanced once in the layout.

```
/* An example containing placement and constraint sections */

MODULE m1;
   TYPE GENERAL;
```

```
   DIMENSIONS 0 0 55 0 55 20 25 20 25 45 0 45;
   IOLIST;
      M1I1 I  0 30 1 METAL2 RESIS 2.58 CAPAC 6.93;
      M1I2 I  0 10 1 METAL2 RESIS 0.16 CAPAC 6.45;
      M1I3 I 40  0 1 METAL2 RESIS 2.03 CAPAC 0.10;
      M1I4 I 40 20 1 METAL2 RESIS 1.99 CAPAC 3.01;
      M1O1 O 15  0 1 METAL2 RESIS 3.12 CAPAC 9.84;
      M1O2 O 55 10 1 METAL2 RESIS 3.76 CAPAC 3.09;
      M1O3 O 10 45 1 METAL2 RESIS 4.21 CAPAC 7.59;
   ENDIOLIST;
   PIN2PINTIME;
      M1I2  M1O2  30  30;
      M1I4  M1O1  15  15;
   ENDPIN2PINTIME;
ENDMODULE;

MODULE m2;
   TYPE GENERAL;
   DIMENSIONS 0 0 25 0 25 45 0 45;
   IOLIST;
      M2I1 I  0 20 1 METAL2 RESIS 3.71 CAPAC 9.60;
      M2I2 I 25 30 1 METAL2 RESIS 2.06 CAPAC 6.06;
      M2I3 I 10 45 1 METAL2 RESIS 1.06 CAPAC 2.71;
      M2O1 O  0 35 1 METAL2 RESIS 4.48 CAPAC 5.68;
      M2O2 O  0  5 1 METAL2 RESIS 2.61 CAPAC 1.93;
      M2O3 O 25 15 1 METAL2 RESIS 4.96 CAPAC 5.81;
   ENDIOLIST;
ENDMODULE;

MODULE m3;
   TYPE GENERAL;
   DIMENSIONS 0 0 30 0 30 25 0 25;
   IOLIST;
      M3I1 I  0 10 1 METAL2 RESIS 2.22 CAPAC 1.24;
      M3I2 I 30 15 1 METAL2 RESIS 4.22 CAPAC 7.51;
      M3I3 I 20 25 1 METAL2 RESIS 2.30 CAPAC 3.66;
      M3I4 I  5 25 1 METAL2 RESIS 4.66 CAPAC 7.97;
      M3O1 O  0 15 1 METAL2 RESIS 0.23 CAPAC 4.26;
   ENDIOLIST;
   PIN2PINTIME;
      M3I3  M3O1  10  10;
   ENDPIN2PINTIME;
ENDMODULE;

MODULE bound;
   TYPE PARENT;
   DIMENSIONS 0 0 120 0 120 80 0 80;
   IOLIST;
```

```
      OUT1 PO  25  0 1 METAL2 RESIS 4.95 CAPAC 7.01;
      IN3  PI  65  0 1 METAL2 RESIS 3.39 CAPAC 7.16;
      OUT2 PO 100  0 1 METAL2 RESIS 2.90 CAPAC 1.51;
      IN4  PI 120 15 1 METAL2 RESIS 2.90 CAPAC 2.97;
      IN5  PI 120 40 1 METAL2 RESIS 3.94 CAPAC 9.10;
      IN6  PI 120 65 1 METAL2 RESIS 0.71 CAPAC 8.06;
      IN7  PI  95 80 1 METAL2 RESIS 2.30 CAPAC 9.95;
      OUT3 PO  70 80 1 METAL2 RESIS 2.47 CAPAC 2.30;
      OUT4 PO  40 80 1 METAL2 RESIS 3.71 CAPAC 2.28;
      IN8  PI  20 80 1 METAL2 RESIS 3.23 CAPAC 9.95;
      IN1  PI  0  55 1 METAL2 RESIS 1.41 CAPAC 3.31;
      IN2  PI  0  25 1 METAL2 RESIS 4.07 CAPAC 2.27;
   ENDIOLIST;
   NETWORK;
      PM1 m1 IN1  IN2  IN3  NET1 OUT1 OUT2 OUT4;
      PM2 m2 NET1 IN5  NET2 OUT3 NET2 NC1;
      PM3 m3 IN1  NET2 IN7  IN8  NET1;
   ENDNETWORK;
   ACTIVITY;
     OUT1 0.05;
     IN3  0.50;
     OUT2 0.02;
     IN4  0.50;
     IN5  0.50;
     IN6  0.50;
     IN7  0.50;
     OUT3 0.33;
     OUT4 0.09;
     IN8  0.50;
     IN1  0.50;
     IN2  0.50;
     NET1 0.23;
     NET2 0.30;
     NC1  0.16;
   ENDACTIVITY;
   PLACEMENT;
     PM1 10.0 10.0 ROT270 RFLY;
     PM2 85.0 10.0;
     PM3 45.0 45.0;
   ENDPLACEMENT;
   CRITICALNETS;
     NET1 25.00;
   ENDCRITICALNETS;
   PIN2PINTIME;
      IN2  OUT2  50 50;
      IN7  OUT1  65 65;
   ENDPIN2PINTIME;
ENDMODULE;
```

A.2.2 Rules and Constraints Specifications

In addition to the LPY circuit description, a file specifying a sub-set of the design rules and constraints for the layout is required. This file is expected to have the same name as the circuit description, with extension .RCF. Its contents should be organized as follows.

```
RULES;
    LAYER <layername> <wirewidth> <wirespacing>
          <conductance> <capacitance>;
    .
    .
    .
ENDRULES;
CONSTRAINTS;
    [ASPECT <aspectratio>;]
    [CORE <height> <width>;]
    [DELAY <maxlength>;]
ENDCONSTRAINTS;
```

The RULES section allow to specify the wire width and wire spacing for each layer in the layout, as well as the sheet conductance and capacitance of each layer. The constraints section allows to optionally impose constraints in the aspect ratio or the core dimensions of the resulting layout. Also, the length of the delay critical net can be optionally specified in this section.

A.3 Output Files

The number and type of output files produced during the execution of the SAPPO program will depend on the command line parameters specified during its invocation.. The files produced by the program will be in the same directory as the input LPY file. Output files will have the same the same name as the input file, with the corresponding extension. An exception to this rule is the output LPY file which will be named "*filename*_o.LPY ". This last file is the one containing the placement produced by the program in its PLACEMENT section. Additionally, a report file with extension RPT will be also created. The output files are formatted in plain ASCII and their contents is self-explanatory.

A.4 Support for Other Netlist Formats

In order to provide a means for processing with the SAPPO layouts specified in netlist formats other than the LPY, a translation tool was developed. This translator, called yal2lpy, allows for converting YAL netlists into the LPY format, provided that the switching activities on input and internal nodes are known. When invoked without parameters, the following output is produced:

```
yal2lpy: Converts YAL files to LPY format.
Usage:
     yal2lpy <infile> [outfile | -]
where,
     infile = Input YAL file (required).
     outfile = Optional output LPY file. If no outfile is
               specified, infile.LPY is created. Using ``-'' sends
               output to the stdout device.
```

The switching activity specifications should be provided in a file with extension ACT, formatted according to the output provided by the SIS power estimation tool.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] H. Mehta, M. Borah, R. M. Owens, and M. J. Irwing. Accurate estimation of combinational circuit activity. In *Proc. 32nd IEEE/ACM Design Aut. Conf.*, pages 618–622, Jun. 1995.

[2] F. Najm, S. Goel, and I. Hajj. Power estimation in sequential circuits. In *Proc. 32nd IEEE/ACM Design Aut. Conf.*, pages 635–640, Jun. 1995.

[3] C.-Y. Tsui, M. Pedram, and A. M. Despain. Exact and approximate methods for calculating signal and transition probabilities in FSMs. In *Proc. Of the 31th. ACM/IEEE Design Aut. Conf.*, pages 18–23, Jun. 1994.

[4] R. Burch, F. N. Najm, P. Yang, and T. Trick. A monte carlo approach for power estimation. *IEEE Transactions on VLSI Systems*, 1(1):63–71, Mar. 1993.

[5] A. Ghosh, S. Devadas, K. Keutzer, and J. White. Estimation of average switching activity in combinational and sequential circuits. In *Proc. 29th IEEE/ACM Design Aut. Conf.*, pages 253–259, Jun. 1992.

[6] F. N. Najm. Transition density, a stochastic measure of activity in digital circuits. In *Proc. Of the 28th ACM/IEEE Design Aut. Conf.*, pages 644–649, Jun. 1991.

[7] S. Iman and M. Pedram. Logic extraction and factorization for low power. In *Proc. 32nd IEEE/ACM Design Aut. Conf.*, pages 248–253, Jun. 1995.

[8] R. San Martin and J. P. Knight. Power-profiler: Optimizing ASICs power consumption at the behavioral level. In *Proc. 32nd IEEE/ACM Design Aut. Conf.*, pages 42–47, Jun. 1995.

[9] L. S. Nielsen, C. Niessen, J. Sparso, and V. VanBerkel. Low-power operation using self-timed circuits and adaptive scaling of the supply voltage. *IEEE Transactions on VLSI Systems*, 2(4):391–397, Dec. 1994.

[10] R. Hossain, M. Zheng, and A. Albicki. Reducing power dissipation in serially connected MOSFET circuits via transistor reordering. In *Proc. IEEE Int. Conf. On Comp. Design*, pages 614–617, Oct. 1994.

[11] V. Tiwari, S. Malik, and A. Wolfe. Power analysis of embedded software: A first step towards software power minimization. *IEEE Transactions on VLSI Systems*, 2(4):437–445, Dec. 1994.

[12] V. Tiwari, P. Ashar, and S. Malik. Technology mapping for low power. In *Proc. 30th IEEE/ACM Design Aut. Conf.*, pages 74–79, Jun. 1993.

[13] K.-Y. Chao and D. F. Wong. Floorplaning for low power designs. In *Proc. Of the 1995 IEEE Int. Symp. On Circuits and Systems*, pages 45–48, May 1996.

[14] H. Vaishanav and M. Pedram. Pcube: A performance driven placement algorithm for low power designs. In *Proc. Of the 1993 European Design Aut. Conf.*, pages 72–77, Sep. 1993.

[15] N. Weste and K. Eshraghian. *Principles of CMOS VLSI Design.* Addison-Wesley Co., Reading, MA, second edition, 1993.

[16] H. Veendrick. Short-circuit dissipation of static CMOS circuitry and its impact on the design of buffer circuits. *IEEE Journal of Solid-State Circuits*, SC-19(4):468–473, Aug. 1984.

[17] J. M. Rabaey and M. Pedram. *Low Power Design Methodologies.* Kluwer Academic Publishers, Norwell, MA 02061, 1996.

[18] S. M. Kang. Accurate simulation of power dissipation in VLSI circuits. *IEEE Journal of Solid-State Circuits*, SC-21(5):889–891, Oct. 1986.

[19] K. P. Parker and E. J. McCluskey. Probabilistic treatment of general combinational networks. *IEEE Transactions on Computers*, C-24:668–674, Jun. 1975.

[20] F. Najm. Transition density: A new measure of activity in digital circuits. *IEEE Transactions on Computer Aided Design*, 12(2):310–323, Feb. 1993.

[21] S. M. Ross. *Introduction to Probability Models.* Academic Press, Inc., San Diego, CA 92101, fifth edition, 1993.

[22] M. Xakellis and F. Najm. Statistical estimation of the switching activity in digital circuits. In *Proc. 31st IEEE/ACM Design Aut. Conf.*, pages 728–733, Jun. 1994.

[23] G. I. Stamoulis. A monte-carlo approach for the accurate and efficient estimation of average transition probabilities in sequential logic circuits. In *Proc. Of the IEEE 1996 Custom IC Conf.*, pages 221–224, May 1996.

[24] L-P. Yuan, C-C. Teng, and S-M. Kang. Nonparametric estimation of average power dissipation in CMOS VLSI circuits. In *Proc. Of the IEEE 1996 Custom IC Conf.*, pages 225–228, May 1996.

[25] M. H. DeGroot. *Probability and Statistics*. Addison-Wesley Pub. Co., Reading, MA 02061, second edition, 1986.

[26] J. Savir, G. S. Ditlow, and P. H. Bardell. Random patern testability. *IEEE Transactions on Computers*, C-33(1):79–90, Jan. 1984.

[27] M. A. Cirit. Estimating dynamic power consumption of CMOS circuits. In *Proc. Of IEEE Int. Conf. On CAD*, pages 534–537, Nov. 1987.

[28] R. E. Bryant. Graph-base algorithms for boolean function manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, Aug. 1986.

[29] F. Najm. Low-pass filter for computing the transition density in digital circuits. *IEEE Transactions on VLSI Systems*, 13(9):1123–1131, Sep. 1994.

[30] J. Monteiro, S. Devadas, and B. Lin. A methodology for efficient estimation of switching activity in sequential logic circuits. In *Proc. 31st IEEE/ACM Design Aut. Conf.*, pages 12–17, Jun. 1994.

[31] C-Y. Tsui, J. Monteiro, M. Pedram, S. Devadas, A. Despain, and B. Lin. Power estimation for sequential circuits. *IEEE Transactions on VLSI Systems*, 3(3):404–416, Sep. 1995.

[32] P. H. Schneider, U. Schlichtmann, and K. J. Antreich. A new power estimation technique with application to decomposition of boolean functions for low power. In *Proc. Of the 1994 European Design Aut. Conf.*, pages 388–393, Sep. 1994.

[33] R. Marculescu, D. Marculescu, and M. Pedram. Switching activity analysis considering spatiotemporal correlations. In *Proc. Of IEEE Int. Conf. On CAD*, pages 294–299, Nov. 1994.

[34] R. Marculescu, D. Marculescu, and M. Pedram. Efficient power estimation for highly correlated input streams. In *Proc. 32nd IEEE/ACM Design Aut. Conf.*, pages 628–634, Jun. 1995.

[35] D. I. Cheng, K-T. Cheng, D. C. Wang, and M. Marek-Sadowska. A new hybrid methodology for power estimation. In *Proc. 33rd IEEE/ACM Design Aut. Conf.*, pages ?–?, Jun. 1996.

[36] D. I. Cheng, M. Marek-Sadowska, and K-T Cheng. Speeding up power estimation by topological analysis. In *Proc. Of the 1995 IEEE Custom IC Conf.*, pages 623–626, May 1995.

[37] T-L. Chou and K. Roy. Statistical estimation of sequential circuit activity. In *Proc. Of IEEE Int. Conf. On CAD*, pages 34–37, Nov. 1995.

[38] T-L. Chou and K. Roy. Estimation of sequential circuit activity considering spatial and temporal correlations. In *Proc. Of IEEE Int. Conf. On Comp. Design*, pages 577–582, Oct. 1995.

[39] A. Chandrakasan, S. Sheng, and R. Brodersen. Low-power CMOS digital design. *IEEE Journal of Solid-State Circuits*, 27(4):473–484, Apr. 1992.

[40] S. Devadas and S. Malik. A survey of optimization techniques targeting low power VLSI circuits. In *Proc. 32nd IEEE/ACM Design Aut. Conf.*, pages 242–247, Jun. 1995.

[41] C.-Y. Tsui, M. Pedram, and A. M. Despain. Technology decomposition and mapping targeting low power dissipation. In *Proc. 30th IEEE/ACM Design Aut. Conf.*, pages 68–73, Jun. 1993.

[42] G. E. Sobelman and D. L. Raatz. Low-power multiplier design using delayed evaluation. In *Proc. Of IEEE Int. Symp. On Circuits and Systems*, pages 1564–1567, May 1995.

[43] A. Chandrakasan, R. Allmon, A. Stratakos, and R. Brodersen. Design of portable systems. In *Proc. Of the IEEE 1994 Custom IC Conf.*, pages 259–266, May 1994.

[44] K. Roy and S. C. Prasad. Circuit activity based logic synthesis for low power reliable operations. *IEEE Transactions on VLSI Systems*, 1(4):503–513, Dec. 1993.

[45] E. Olson and S. M. Kang. State assignment for low-power FSM synthesis using genetic local search. In *Proc. Of the IEEE 1994 Custom IC Conf.*, pages 140–143, May 1994.

[46] G. De Micheli. *Synthesis and Optimization of Digital Circuits*. McGraw-Hill, Inc., New York, NY, 1994.

[47] M. Alidina, J. Monteiro, S. Devadas, A. Ghosh, and M. Papaefthymiou. Precomputation-based sequential logic optimization for low power. *IEEE Transactions on VLSI Systems*, 2(4):426–436, Dec. 1994.

[48] A. Raghunathan and N. Jha. Behavioral synthesis for low power. In *Proc. IEEE Int. Conf. On Comp. Design*, pages 318–322, Oct. 1994.

[49] L. Lavagno, P. McGeer, A. Saldanha, and A. Sangiovanni-Vicentelli. Timed Shannon circuits: A power-efficient design style and synthesis tool. In *Proc. 32nd IEEE/ACM Design Aut. Conf.*, pages 254–260, Jun. 1995.

[50] A. Vittal and M. Marek-Sadowska. Power optimal buffered clock tree design. In *Proc. 32nd IEEE/ACM Design Aut. Conf.*, pages 497–502, Jun. 1995.

[51] J. G. Xi and W. Dai. Buffer insertion and sizing under process variations for low power clock distribution. In *Proc. 32nd IEEE/ACM Design Aut. Conf.*, pages 491–496, Jun. 1995.

[52] S. Sahni and A. Bhatt. The complexity of design automation problems. In *Proc. Of the 17th IEEE/ACM Design Aut. Conf.*, pages 402–411, Jun. 1980.

[53] W. Donath. Complexity theory and design automation. In *Proc. Of the 17th IEEE/ACM Design Aut. Conf.*, pages 412–419, Jun. 1980.

[54] S. M. Sait and H. Youssef. *VLSI Physical Design Automation.* IEEE Press, Inc., Piscataway, NJ 08855, 1995.

[55] N. Sherwani. *Algorithms for VLSI Physical Design Automation.* Kluwer Academic Publishers, Norwell, MA 02061, second edition, 1997.

[56] E. Reingold and K. Supowit. Hierarchy-driven amalgamation of standard and macro cells. *IEEE Transactions on Computer Aided Design*, CAD-3(1):3–11, Jan. 1984.

[57] M. Upton, K. Samii, and S. Sugiyama. Integrated placement for mixed macro cell and standard cell designs. In *Proc. Of the 27th. IEEE/ACM Design Aut. Conf.*, pages 482–485, Jun. 1990.

[58] G.Chartrand and O. Oellermann. *Applied and Algorithmic Graph Theory.* McGraw Hill, Inc., Hightstown, NJ 08520, 1993.

[59] M. A. Jimenez and M. Shanblatt. Median biased steiner tree heuriscs in the rectilinear plane for low-power physical layout. In *Proc. Of the 41st Midwest Symp. On Circuits and Systems*, pages 268–271, Aug. 1998.

[60] F. Hwang. On Steiner minimal trees with rectilinear distance. *SIAM J. of Applied Math.*, 30(1):104–114, Jan. 1976.

[61] F. Hwang, D. Richards, and P. Winter. *The Steiner Tree Problem.* Elsevier Science Publishers B.V., AE Amsterdam, The Netherlands, 1992.

[62] A. Kahng and G. Robins. A new class of iterative Steiner tree heuristics with good performance. *IEEE Transactions on Computer Aided Design*, 11(7):893–902, Jul. 1992.

[63] M. Hanan. On Steiner's problem with rectilinear distance. *SIAM J. of Applied Math.*, 14(2):255–265, 1966.

[64] P. Winter. Reductions for the rectilinear Steiner tree problem. Technical Report RRR 11-95, Rutcor Research Report, Mar. 1995.

[65] M. Garey and D. Johnson. The rectilinear Steiner problem is NP-complete. *SIAM J. of Applied Math.*, 32(4):826–834, 1977.

[66] J. Salowe and D. Warme. An exact rectilinear steiner tree algorithm. In *Proc. Of the IEEE Int. Conf. On Comp. Design*, pages 472–475, Oct. 1993.

[67] J. Griffith, G. Robins, J. Salowe, and T. Zhang. Closing the gap: Near-optimal Steiner trees in polynomial time. *IEEE Transactions on Computer Aided Design*, 13(11):1351–1365, Nov. 1994.

[68] N. Hasan, G. Vijayan, and C. K. Wong. A neighborhood improvement algorithm for rectilinear Steiner trees. In *Proc. Of the 1990 Int. Symp. On Circuits and System*, pages 2869–2872, May 1990.

[69] D. Richards. Fast heuristic algorithms for rectilinear Steiner trees. *Algorithmica*, 1989(4):191–207, Apr. 1989.

[70] J. L. Ganley and J. P. Cohoon. Routing a multi-terminal critical net: Steiner tree construction in the presence of obstacles. In *Proc. Of the 1994 Int. Symp. On Circuits and Systems*, pages 113–116, May 1994.

[71] H. Takahashi and A. Matsuyama. An approximate solution for the Steiner problem in graphs. *Math. Japonica*, 1980(6):573–577, 1980.

[72] L. Kou, G. Markowsky, and L. Berman. A fast algorithm for Steiner trees. *Acta Informatica*, 1981(15):141–145, 1981.

[73] M Diane and J. Plesnik. Three new heuristics for the Steiner problem in graphs. *Acta Math. Univ. Comenianae*, 60(1):105–121, 1991.

[74] P. Winter and J. Smith. Path-distance heuristics for the Steiner. *Algorithmica*, 1992(7):309–327, 1992.

[75] M. Breuer. A class of min-cut placement algorithms. In *Proc. Of the 14th IEEE/ACM Design Aut. Conf.*, pages 284–290, Jun. 1977.

[76] U. Lauther. A min-cut placement algorithm for general cell assemblies based on a graph representation. In *Proc. Of the 16th IEEE/ACM Design Aut. Conf.*, pages 1–10, Jun. 1979.

[77] A. Dunlop and B. Kernighan. A procedure for placement of standard cell VLSI circuits. *IEEE Transactions on Computer Aided Design*, CAD-4(1):92–98, Jan. 1985.

[78] P. Suaris and G. Kedem. Quadrisection: A new approach to standard cell layout. In *Proc. Of the IEEE Int. Conf. On Computer Design*, pages 474–477, 1987.

[79] K. Antreich, F. Johannes, and F. Kirsh. A new approach for solving the placement problem using force models. In *Proc. Of the IEEE Int. Symp. On Circuits and Systems*, pages 481–486, 1982.

[80] L. Sha and R. Dutton. An analytical algorithm for placement of arbitrarily sized rectangular blocks. In *Proc. Of the 22nd IEEE/ACM Design Aut. Conf.*, pages 602–607, Jun. 1985.

[81] S. Goto. An efficient algorithm for the two-dimensional placement problem in circuit layout. *IEEE Transactions on Circuits and Systems*, CAS-28:12–18, Jan. 1981.

[82] N. Quinn and M. Breuer. A force directed component placement procedure for printed circuit boards. *IEEE Transactions on Circuit and Systems*, CAS-26:377–388, Jun. 1979.

[83] K. Shahookar and P. Mazumder. VLSI cell placement techniques. *ACM Computing Surveys*, 23(2):143–220, Jun. 1991.

[84] B. Buckles and F. Petry. *Genetic Algorithms*. IEEE Computer Society Press, Los Alamitos, CA 90720, 1992.

[85] J. Cohoon and W. Paris. Genetic placement. In *Proc. Of the IEEE Int. Conf. On Computer Aided Design*, pages 422–486, 1986.

[86] K. Shahookar and P. Mazumder. A genetic approach to standard cell placement using meta-genetic parameter optimization. *IEEE Transactions on Computer Aided Design*, 9(5):500–511, May 1990.

[87] H. Chan, P. Mazumder, and K. Shahookar. Macro-cell and module placement by genetic adaptive search with bitmap represented chromosome. *Integration: The VLSI Journal*, 12(1):49–77, Nov. 1991.

[88] R. Kling and P. Banerjee. Empirical and theoretical studies on the simulated evolution method applied to standard cell placement. *IEEE Transactions on Computer Aided Design*, 10(10):1303–1315, Oct. 1991.

[89] S. Kirkpatrik, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, May 1983.

[90] Emilie Aarts and Jan Korst. *Simulated Annealing and Bolzmann Machines*. John Wiley and Sons, Inc., New York, NY, 1989.

[91] N. Metropolis, M. Rosenbluth, A. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[92] Peter Van Laarhoven and Emile Aarts. *Simulated Annealing: Theory and Applications*. D. Reidel Publishing Co., Dordrecht, Holland, 1987.

[93] C. Sechen. Chip-planning, placement, and global routing of macro-cell integrated circuits using simulated annealing. *International Journal of Computer Aided VLSI Design*, 2:127–158, 1990.

[94] S. White. Concepts of scale in simulated annealing. In *Proc. Of the IEEE Int. Conf. On Computer Design*, pages 646–651, 1984.

[95] M. Huang, F. Romeo, and A. Sangivanni-Vicentelli. An efficient general cooling schedule for simulated annealing. In *Proc. Of the IEEE Int. Conf. On Computer Aided Design*, pages 381–384, Nov. 1986.

[96] J. Lam and M. Delosme. Performance of a new annealing schedule. In *Proc. Of the 25th IEEE/ACM Design Aut. Conf.*, pages 306–311, Jun. 1988.

[97] K. Boese, A. Kahng, and C. Albert. Best-so-far vs. where-you-are: New perspectives on simulated annealing for cad. In *Proc. Of the European Design Aut. Conf.*, pages 78–83, Sep. 1993.

[98] C. Sechen and A. Sangiovanni-Vicentelli. The TimberWolf placement and routing package. *IEEE Journal of Solid State Circuits*, SC-20(2):510–522, Apr. 1985.

[99] W. Swartz and C. Sechen. New algorithms for the placement and routing of macro cells. In *Proc. Of the IEEE Int. Conf. On Computer Aided Design*, pages 336–339, 1990.

[100] W. P. Swartz. *Automatic Layout of Analog and Digital Mixed Macro/Standard Cell Integrated Circuits*. PhD thesis, Yale University, May 1993.

[101] P. Bannerjee and M. Jones. A parallel simulated annealing algorithm for standard cell placement on a hypercube computer. In *Proc. Of the IEEE Int. Conf. On Computer Design*, pages 34–39, 1986.

[102] M. Yu. A study of the applicability of hopfield decision neural nets to VLSI CAD. In *Proc. Of The 26th ACM/IEEE Design Aut. Conf.*, pages 412–417, Jun. 1989.

[103] M. Takahashi, K. Kyuma, and E. Funada. 10000 cell placement optimization using a self-organization map. In *Proc. Of the 1993 Int. Joint Conf. On Neural Networks*, pages 2417–2420, 1993.

[104] R. Chang and P. Hsiao. Arbitrarily sized cell placement by self-organizing neural networks. In *Proc. Of the Int. Symp. On Circuits and Systems*, pages 2043–2046, 1993.

[105] R. Lin and E. Shragowitz. Fuzzy logic approach to placement problem. In *Proc. Of the 29th Design Aut. Conf.*, pages 153–158, Jun. 1992.

[106] C. Ball, A. Just, and D. Mlynski. A fuzzy mean field approach for partitioning and placement. In *Proc. Of the Int. Symp. On Circuits and Systems*, pages 373–376, 1995.

[107] E. Barke. Line-to-ground capacitance calculation for VLSI: A comparison. *IEEE Transactions on Computer Aided Design*, 7(2):295–298, Feb. 1988.

[108] F. Brglez, D. Bryan, and K. Kominski. Combinational profiles of sequential benchmark circuits. In *Proc. Of the IEEE Int. Symposium On Circuits and Systems*, pages 1929–1934, May 1989.

[109] J. P. Cohoon and D. S. Richards. Optimal two-terminal $\alpha-\beta$ wire routing. *Integration: The VLSI Journal*, 6(1):35–57, May 1988.

[110] J. L. Bentley and T. A. Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Transactions on Computers*, C-28(9):643–647, Sep. 1979.

[111] A. Tenenbaum, Y. Langsam, and M. Augenstein. *Data Structures Using C*. Prentice Hall, Inc., Englewood Cliffs, NJ 07632, 1990.

[112] C. Sechen. *VLSI Placement and Routing using Simulated Annealing*. Kluwer Academic Publishers, Norwell, MA 02061, 1988.

[113] H. Onodera, Y. Taniguchi, and K. Tamaru. Branch-and-bound placement for building block layout. In *Proc. Of the 28th IEEE/ACM Design Aut. Conf.*, pages 433–439, Jun. 1991.

[114] H. Cai. Connectivity biased channel construction and ordering for building-block layout. In *Proc. Of the 25th IEEE/ACM Design Aut. Conf.*, pages 560–565, Jun. 1988.

[115] T. Gao, P. M. Vaidya, and L. Liu. A performance driven macro-cell placement algorithm. In *Proc. Of the IEEE Int. Conf. On Computer Aided Design*, pages 147–152, Nov. 1992.

[116] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani. Rectangle-packing-based module placement. In *Proc. Of the IEEE Int. Conf. On Computer Aided Design*, pages 472–479, Nov. 1995.

[117] J. Bentley and D. Wood. An optimal worst case algorithm for reporting intersections of rectangles. *IEEE Transactions on Computers*, C-29(7):571–577, Jul. 1980.

[118] W. Lipski and F. Preparata. Finding the contour of a union of iso-oriented rectangles. *Journal of Algorithms*, 1(3):235–246, Sep. 1980.

[119] R. Guting. An optimal contour algorithm for iso-oriented rectangles. *Journal of Algorithms*, 5:303–326, 1984.

[120] H. Stark and J. Woods. *Probability, Random Processes, and Estimation Theory for Engineers*. Prentice Hall, Inc., Englewood Cliffs, NJ 07632, second edition, 1994.

[121] B. Eschermann, W. Dai, E. Kuh, and M. Pedram. Hierarchical placement for macro-cells: A "meet in the middle" approach. In *Proc. Of the IEEE Int. Conf. On CAD*, pages 460–463, Nov. 1988.

[122] L. W. Hagen, D. J. Huang, and A. B. Kahng. Quantified suboptimality of VLSI layout heuristics. In *Proc. Of the 32nd IEEE/ACM Design Aut. Conf.*, pages 216–221, Jun. 1995.

[123] K. Keutzer and P. Vanbekbergen. Impact of CAD on the design of low power digital circuits. In *Proc. Of the 1994 IEEE Symp. On Low Power Electronics*, pages 42–45, Oct. 1994.